1973

The generation of gamma distributed variates and an investigation of a trend test for the gamma renewal process.

Robinson, David Walter.

THE GENERATION OF GAMMA DISTRIBUTED VARIATES AND AN INVESTIGATION OF A TREND TEST FOR THE GAMMA RENEWAL PROCESS

David Walter Robinson

THE GENERATION OF GAMMA DISTRIBUTED VARIATES AND AN INVESTIGATION OF A TREND TEST FOR THE GAMMA RENEWAL PROCESS

by

David Walter Robinson

Thesis Advisor: P.A.W. Lewis

June 1973

The Generation of Gamma Distributed Variates and an Investigation of a Trend Test for the Gamma Renewal Process

by

David Walter Robinson
Lieutenant, United States Navy
B.S., United States Naval Academy, 1965

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 1973

ABSTRACT

In testing the hypothesis that there is no monotone trend in a gamma renewal process, the use of the statistic

$$Y_J = Y_{2J} / Y_{1J},$$

where

$$Y_{1J} = \sum_{i=1}^{J} X_i$$

and

$$Y_{2J} = \sum_{i=1}^{J} S_i,$$

is investigated. The mean and variance of Y_J is developed as a function of J and it is shown that Y_J is asymptotically normal as $J \to \infty$ for the gamma renewal process. A high-speed, theoretically exact gamma pseudo-random variate generator is developed, tested and compared with other known techniques. The generator is then used to obtain the distribution of Y_J through digital computer simulation for small and moderate values of J.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

The assistance and guidance provided by my thesis advisor, Prof. P.A.W. Lewis, is gratefully acknowledged. Additionally, the support and aid given by the Naval Postgraduate School W.R. Church Computer Center and its staff, especially Mr. G.P. Learmonth, was very helpful in completing this work. Finally, thanks are due Prof. G.L. Barksdale, Jr. for his development of the Text Processing System used to reproduce this document.

I. INTRODUCTION

A renewal process is a model for a series of events which occur at random times; the times between each two successive events are independent and identically distributed positive random variables. The time between event i-1 and event i is denoted by X_i, i=1,2,..., while the random variable S_i represents the time to the i-th event. When zero is taken as the time of initiation of the series of events then

$$S_i = \sum_{n=1}^{i} X_n \qquad (i=1,2,\ldots).$$

An equivalent representation of a series of events is the counting function N(t) which represents the number of events occurring in the interval (0,t]. The equivalence of the two representations is expressed in the fundamental identity

$$N(t) < n \text{ if and only if } S_n > t \qquad (n=1,2,\ldots).$$

Thus $P\{N(t) < n\} = P\{S_n > t\}$, where the notation $P\{\ \}$ represents the probability of the event within the brackets.

Two important characteristics of any series of events are the so-called renewal function M(t), which is the expected number of events up to time t, and its derivative the rate function:

$$M(t) = E[N(t)],$$

$$\lambda(t) = \frac{d}{dt} M(t).$$

A stationary renewal process is one in which N(t) has stationary increments (see Cox and Lewis [6], Chapter 4, for details). In this case, the renewal function is

$$M(t) = t / u,$$

and $\lambda(t) = 1/u$, where u is the mean time between events.

For an arbitrary series of events there are many alternatives to the renewal process. Successive inter-event times may not be independent or may depend on the entire history of the process, or the distributions of the times between events or the increments of N(t) may change in some manner as the process continues. When these distributional changes involve the expected values of the inter-event times or the value of $\lambda(t)$, then a trend is said to exist in the process. The trend may be on the serial number of the event; for example, a radio set may tend to fail more frequently as a function of the total number of failures which have occurred. Alternatively, the trend may be essentially independent of the number of events and reflect basically the elapsed time t; such a trend on time might be observed in the arrival of jobs at a computer facility over a 24-hour day where the job arrival rate depends mainly on the time of day and not on the number of jobs submitted. A trend on time is usually expressed through a non-linear time dependence in M(t).

Trends may be increasing, decreasing, cyclic or some combination of the three. The various statistical procedures designed to test whether an arbitrary series of events has a trend of any type are quite specific as to the type of process and the trend model assumed. For a more complete discussion of tests for trend in series of events see Cox and Lewis [6], Cox [7] and Lewis [18].

The simplest form of renewal process is the Poisson process in which the inter-event times have the negative exponential distribution. A more general process is the

gamma renewal process in which the X_i's are independent gamma random variables with density function

$$f(x) = \frac{1}{\Gamma(k)} \lambda^{k} x^{k-1} e^{-\lambda x} \qquad (k > 0).$$

In this parameterization, λ is a scale parameter while k characterizes the distribution and is called the shape parameter. The function $\Gamma(k)$ is the complete gamma function or integral

$$\Gamma(k) = \int_0^{\infty} x^{k-1} e^{-x} dx .$$

In testing for trend on serial number in the gamma renewal process the use of the statistic

$$Y_J = Y_{2J} / Y_{1J},$$

where

$$Y_{1J} = \sum_{i=1}^{J} X_i$$

and

$$Y_{2J} = \sum_{i=1}^{J} S_i$$

is investigated in this paper. A test for trend using Y_J is developed, and the distribution of Y_J is investigated through simulation for small values of J and through normal theory results as $J \rightarrow \infty$.

II. THEORETICAL RESULTS

A. BACKGROUND

Cox [5] proposed the use of the statistic

$$B = \sum_{i=1}^{J} t_i$$

as a test for trend in a Poisson process, where $t_1, t_2, \ldots, t_J$ are the normalized times to events in a fixed observation interval of length T and J is the number of events occurring in (0,T], that is, the observed value of N(T). Bartholomew [4] dealt with the statistic

$$m = \frac{1}{J} B$$

and developed results on the asymptotic relative efficiency of tests for trend based on this statistic against several different trend models. Lewis [18] showed that the test based on m is not a consistent test for stationary Poisson processes against stationary gamma renewal alternatives, although it had been used for that purpose.

The test based on the statistic B is a conditional test, the conditioning being on the observed value J of the random variable N(T). This is because the test is designed to detect trend regardless of the rate of occurrence of events. The test arises from the non-homogeneous Poisson process with rate function

$$\lambda(t) = \exp\{a + bt\} = \lambda e^{bt}.$$

In this case it is desired to test whether b=0, that is whether there is a trend. It is possible to obtain the likelihood function $L(\lambda, b)$ of the observations for this model and from Neyman-Pearson theory conclude that the test based on B is the uniformly most powerful conditional test

$$H_0: b=0, \lambda > 0$$

$$H_1: b \neq 0, \lambda > 0$$

for every value of T. For details, see Cox and Lewis [6], chapters 2 and 3.

For the foregoing model, the parameter λ is a nuisance parameter since it has no effect on the trend; it merely controls the base rate of the process. It is eliminated in the B test by conditioning on $J = N(T)$, this being a sufficient statistic for λ for any fixed value of b.

There is an analogous situation when testing for trend on serial number and the process is observed for a fixed number of intervals J. Here, the trend model is one of exponentially distributed intervals with

$$E[X_i] = \exp\{a + bi\} = \lambda e^{bi} \qquad (i=1,2,\ldots).$$

The sufficient statistic for λ is now $S_J = Y_{1J}$ so that the test is now conditioned on the observed value of Y_{1J} and the statistic analogous to B is

$$Y_{2J} = \sum_{i=1}^{J} S_i = \sum_{i=1}^{J} (J+1-i)\, X_i.$$

Asymptotically, there is no difference between conditioning on a fixed time interval and conditioning on a fixed number of events but the latter turns out to be more convenient for the gamma renewal process. In this case

performing a test for trend analogous to the Poisson process test based on B requires the determination of the conditional distribution of Y_{2J} given Y_{1J}.

B. THE GAMMA RENEWAL PROCESS

In the gamma renewal process, it is both analytically and computationally simpler to test for trend on serial number over a fixed number of observations than to test for trend in time over a fixed time period. The computational advantage of the former test arises from the ease of computer simulation of a fixed number of events as compared to the difficulties inherent in continuous time simulation control. Analytically, the gamma renewal process has certain theoretical properties which simplify consideration of the serial trend test.

For the gamma renewal process with shape parameter k and the assumed trend model (Cox [7])

$$E[X_i] = \exp\{a + bi\} = \lambda e^{bi} \qquad (i=1,2,\ldots)$$

it is possible to set up the likelihood function $L(k,b,\lambda)$. Explicit results, however, cannot be obtained for estimating b or testing whether b=0; it is therefore usual to use the Poisson test based on B in this case. The test will probably be relatively powerful for k near one but not when k is small as it uses very little information about the individual X_i's. In fact, the test is based only on the centroid of the times to events and not on times between events.

Elimination of the nuisance parameter λ cannot be accomplished for the gamma renewal process by conditioning

on S_J as it could in the Poisson case, since S_J is not a sufficient statistic for λ for all values of b. However, since λ is a pure scale factor the ratio statistic

$$Y_J = Y_{2J} / Y_{1J}$$

is free of λ for any b and k and is thus a reasonable alternative statistic on which to base a test which is free of the scale factor.

The distributions of the two test statistics (conditional and ratio) are equivalent in the gamma case under the null hypothesis (b=0) because of a result which characterizes the gamma distribution. Laha [16] and Lukacs [21] have established that, for a sequence of independent and identically distributed random variables $X_1, X_2, \ldots, X_n$, the statistics

$$\sum_{i=1}^{J} (J+1-i) X_i \; / \; \sum_{i=1}^{J} X_i$$

and

$$\sum_{i=1}^{J} X_i$$

are independent if and only if the X_i's have the gamma distribution. Thus, Y_J is independent of Y_{1J} for the gamma renewal process. Using this result,

$$P\{Y_J \leq z\} = P\{Y_{2J} / Y_{1J} \leq z\}$$

$$= P\{Y_{2J} / Y_{1J} \leq z \mid Y_{1J} = y\}$$

$$= P\{Y_{2J} \leq zy \mid Y_{1J} = y\},$$

so that the conditional and ratio statistics have the same distribution. Furthermore, the gamma renewal process is the only one for which this is the case.

A further result of the above characterization and the probability relationship is that, if the joint distribution of Y_{1J} and Y_{2J} is asymptotically bivariate normal (a result which is proved in the following Section), then Y_J must also be asymptotically normal. This follows since the conditional distribution of Y_{2J} given Y_{2J} will be asymptotically normal under these conditions. Thus, although the asymptotic distribution of a ratio statistic such as Y_J can be very difficult to obtain in general, an expression can be found for the gamma case.

Trend tests using Y_J for other types of renewal process could and should be considered but the gamma process was chosen deliberately to take advantage of the above simplifying distributional results. The gamma process would probably not be the simplest case to consider under the trend alternative, however.

In the succeeding sections, results on the asymptotic distribution of Y_J and the mean and variance of Y_J for small values of J are developed. In addition, a description of a computer simulation experiment which was performed to obtain the distribution of Y_J for small values of J is described and its results presented. The power of the Y_J test for the gamma process has not been investigated nor has its power relative to nonparametric tests or tests using data

transformations been examined.

C. ASYMPTOTIC DISTRIBUTIONS

The statistics Y_{1J} and Y_{2J} for a renewal process are defined by

$$Y_{1J} = \sum_{i=1}^{J} X_i ,$$

$$Y_{2J} = \sum_{i=1}^{J} S_i = \sum_{i=1}^{J} (J+1-i) \; X_i ,$$

where $X_1, X_2, \ldots, X_J$ are the process inter-event times and are assumed to have finite mean u and finite variance σ^2.

Obviously, Y_{1J} and Y_{2J} are dependent random variables with respective means and variances

$$\begin{aligned} m_1 &= E[Y_{1J}] \\ &= E[\sum_{i=1}^{J} X_i] \\ &= \sum_{i=1}^{J} E[X_i] \\ &= J \; E[X] \\ &= J \; u. \end{aligned}$$

$$\begin{aligned} s_1^2 &= Var[Y_{2J}] \\ &= Var[\sum_{i=1}^{J} X_i] \\ &= \sum_{i=1}^{J} Var[X_i] \\ &= J \; Var[X] \end{aligned}$$

$$= J\sigma^2.$$

$$m_2 = E[Y_{2J}]$$

$$= E[\sum_{i=1}^{J} (J+1-i)\ X_i]$$

$$= \sum_{i=1}^{J} (J+1-i)\ E[X_i]$$

$$= E[X] \cdot [\sum_{i=1}^{J} (J+1) - \sum_{i=1}^{J} i]$$

$$= \frac{J(J+1)}{2} u.$$

$$s_2^2 = Var[Y_{2J}]$$

$$= Var[\sum_{i=1}^{J} (J+1-i)\ X_i]$$

$$= \sum_{i=1}^{J} (J+1-i)^2\ Var[X_i]$$

$$= Var[X] \sum_{i=1}^{J} i^2$$

$$= \frac{J(J+1)(2J+1)\sigma^2}{6}.$$

To find the correlation coefficient for the statistics Y_{1J} and Y_{2J} it is necessary to determine the first joint moment:

$$m_{12} = E[Y_{1J} Y_{2J}]$$

$$= E[\sum_{i=1}^{J} X_i \cdot \sum_{i=1}^{J} (J+1-i)\ X_i]$$

$$= E[\sum_{i=1}^{J} \sum_{k=1}^{J} (J+1-k)\ X_i X_k]$$

$$m_{12} = E\Big[\sum_{i=1}^{J} (J+1-i)\ X_i^2 + \sum_{i=1}^{J-1} \sum_{k=i+1}^{J} (2J+2-i-k)\ X_i X_k \Big]$$

$$= \sum_{i=1}^{J} (J+1-i)\ E[X_i^2] + \sum_{i=1}^{J-1} \sum_{k=i+1}^{J} (2J+2-i-k)\ E[X_i X_k]$$

$$= E[X^2] \sum_{i=1}^{J} i + \{E[X]\}^2 \sum_{i=1}^{J-1} \sum_{k=i+1}^{J} (i+k) .$$

After some simplification this becomes

$$m_{12} = \frac{J(J+1)}{2} E[X^2] + \frac{J(J^2-1)}{2} u^2 .$$

Then the covariance is

$$u_{11} = E[(Y_{1J} - m_1)(Y_{2J} - m_2)]$$

$$= m_{12} - m_1 m_2$$

$$= \frac{J(J+1)}{2} \sigma^2$$

and the correlation coefficient is

$$\rho = u_{11} / s_1 s_2$$

$$= \frac{J(J+1)}{2} \sigma^2 \Big/ \sqrt{J} \cdot \sqrt{\frac{J(J+1)(2J+1)}{6}}\ \sigma^2$$

$$= \frac{\sqrt{6}}{2} \sqrt{\frac{J+1}{2J+1}} .$$

To obtain asymptotic results for Y_J as $J \to \infty$, it is necessary to show that the joint distribution of Y_{1J} and Y_{2J} is asymptotically a bivariate normal distribution. To do

this it is necessary to apply the Central Limit Theorem to the two-dimensional vector random variable

$$Z_J = (Y_{1J} \; Y_{2J}) .$$

Wald and Wolfowitz [29] have shown that Z_J is asymptotically normal if and only if

$$L_J = l_1 Y_{1J} + l_2 Y_{2J}$$

is also asymptotically normal for every choice of l_1 and l_2.

To apply the Central Limit Theorem to L_J, define

$$\begin{aligned} L'_J &= L_J - E[L_J] \\ &= l_1 \sum_{i=1}^{J} X_i + l_2 \sum_{i=1}^{J} (J+1-i) \; X_i - E[L_J] \\ &= \sum_{i=1}^{J} [l_1 + l_2 (J+1-i)] \; (X_i - u) \\ &= \sum_{i=1}^{J} a_i (X_i - u) . \end{aligned}$$

$$a_i = l_1 + l_2 (J+1-i) .$$

$$F_i(v) = P\{a_i (X_i - u) \leq v\} .$$

$$\begin{aligned} s_1^2 &= Var[L_J] \\ &= Var[L'_J] \\ &= Var[X_i - u] \sum_{i=1}^{J} a_i^2 \end{aligned}$$

$$= \sigma^2 \sum_{i=1}^{J} a_i^2 .$$

Then L_J will be asymptotically normal if and only if the Lindeberg condition is satisfied for every $e > 0$ (Cramer [8]):

$$(2\text{-}1) \qquad \lim_{J \to \infty} \frac{1}{s_1^2} \sum_{i=1}^{J} \left\{ \int_{|v| > e s_1} v^2 dF_i(v) \right\} = 0.$$

After making the substitution $v = a_i z$ the integral becomes

$$\int_{|v| > e s_1} v^2 dF_i(v) = \int_{|a_i z| > e s_1} a_i^2 z^2 dF_i(a_i z).$$

This can be simplified by first noting that

$$\begin{aligned} F_i(a_i z) &= P\{a_i (X_i - u) \leq a_i z\} \\ &= P\{X_i - u \leq z\} \\ &= P\{X_i \leq z+u\} \\ &= F(z+u), \end{aligned}$$

where $F(x)$ is the common distribution function of the X_i's. Secondly, if

$$|a_i z| > e s_1$$

then

$$\begin{aligned} |z| &> e s_1 / |a_i| \\ &> e s_1 / |l_1 + l_2 (J+1-i)| \end{aligned}$$

$$> e\, s_1 / (|l_1| + |l_2(J+1-i)|)$$

$$> e\, s_1 /(|l_1| + |l_2|J).$$

We can thus assume, without loss of generality, that $l_1>0$ and $l_2>0$. Then,

$$\int_{|v|>es_1} v^2 dF_i(v) \leq a_i^2 \int_{|z|>es_1/(l_1+l_2J)} z^2 dF(z+u).$$

$$\frac{1}{s_1^2} \sum_{i=1}^{J} \left\{\int_{|v|>es_1} v^2 dF_i(v)\right\}$$

$$\leq \frac{1}{s_1^2} \left[\sum_{i=1}^{J} a_i^2\right] \int_{|z|>es_1/(l_1+l_2J)} z^2 dF(z+u)$$

$$\leq \frac{1}{\sigma^2} \int_{|z|>es_1/(l_1+l_2J)} z^2 dF(z+u). \tag{2-2}$$

Since by hypothesis the X_i's have a finite second moment and since

$$\lim_{J\to\infty} es_1/(l_1+l_2J)$$

$$\geq (e/l_2) \lim_{J\to\infty} (s_1/J)$$

$$\geq (e\sigma/l_2) \lim_{J\to\infty} \frac{1}{J} \left\{\sum_{i=1}^{J} a_i^2\right\}^{1/2}$$

$$\geq (e\sigma/l_2) \lim_{J\to\infty} \left\{\sum_{i=1}^{J} \frac{1}{J^2} [l_1+l_2(J+1-i)]^2\right\}^{1/2}$$

$$\geq \infty$$

the integral (2-2) approaches zero and so the Lindeberg condition (2-1) is satisfied.

It has thus been shown that Y_{1J} and Y_{2J} are asymptotically jointly normal. From the known properties of the two-dimensional normal distribution (Gnedenko [10]), the conditional distribution of Y_{2J} given Y_{1J} must also be asymptotically normal with mean and variance

$$E[Y_{2J} | Y_{1J} = y_1] = m_2 + (\rho s_2 / s_1)(y_1 - m_1)$$

$$= \frac{J(J+1)}{2} u + \frac{(J+1)}{2} (y_1 - J u)$$

$$= \frac{J+1}{2} y_1 .$$

$$Var[Y_{2J} | Y_{1J} = y_1] = s_2^2 (1 - \rho^2)$$

$$= \frac{J(J+1)(2J+1)}{6} \sigma^2 \left(1 - \frac{6}{4} \frac{J+1}{2J+1} \right)$$

$$= \frac{J(J^2-1)}{12} \sigma^2 .$$

From the results of the previous Section, Y_J is also asymptotically normal with mean and variance

$$E[Y_J] = E[Y_{2J} / Y_{1J} \mid Y_{1J} = y_1]$$

$$= \frac{J+1}{2} .$$

$$Var[Y_J] = Var[Y_{2J} / Y_{1J} \mid Y_{2J} = y_1]$$

$$= \frac{J(J^2-1)}{12} (\sigma / y_1)^2 .$$

The foregoing result then suggests the following procedure to test for trend in a gamma renewal process. The

observed values of Y_{1J} and Y_{2J} can be used to accept or reject the null hypothesis of a trend-less process at any desired significance level α using the conditional normal distribution given above. Rejection of the no-trend hypothesis will take place if the observed value of Y_J is greater than the $1-\alpha/2$ quantile of the distribution of Y_J (for increasing trend) or if Y_J is less than the $\alpha/2$ quantile (decreasing trend).

For the distribution of the sum of a series of independent exponential random variables to become approximately normal, however, requires over 20 terms; the gamma renewal process with shape parameter less than one requires several terms just to produce a single exponential variable. Thus, on the order of 100 events could be required to use the asymptotic test. For a small sample (roughly, one with fewer than $20/k$ events) it may not be possible to use a normal approximation to the distribution of the test statistic.

If sample size is so small that the asymptotic result cannot be used, it is then necessary to develop the distribution of Y_J in order to use this test for trend. If the distribution cannot be determined analytically, computer simulation must be resorted to; it is still possible, however, to use the probability relationship between the distribution of Y_J and Y_{2J} given Y_{1J} to obtain the moments of Y_J.

D. THE MEAN AND VARIANCE OF Y_J FOR SMALL SAMPLES

From the foregoing, it can be concluded that in a gamma renewal process

$$E[Y_J] = E[Y_{2J} / Y_{1J}]$$

$$= E[Y_{2J}] / E[Y_{1J}]$$

since Y_{2J}/Y_{1J} is independent of Y_{1J}. Thus,

$$E[Y_J] = m_2 / m_1$$

$$= \frac{J+1}{2}.$$

Similarly,

$$E[Y_J^2] = E[Y_{2J}^2 / Y_{1J}^2]$$

$$= E[Y_{2J}^2] / E[Y_{1J}^2]$$

$$= (s_2^2 + m_2^2) / (s_1^2 + m_1^2)$$

$$= \frac{(1/6)J(J+1)(2J+1)\sigma^2 + (1/4)J^2(J+1)^2u^2}{J\sigma^2 + J^2u^2}$$

$$= \frac{(J+1)}{12} \frac{2(2J+1)\sigma^2 + 3J(J+1)u^2}{\sigma^2 + Ju^2}.$$

Introducing the mean and variance of the gamma random variable,

$$u = k / \lambda$$

$$\sigma^2 = k / \lambda^2$$

$$E[Y_J^2] = \frac{J+1}{12} \frac{2(2J+1)(k/\lambda^2) + 3J(J+1)(k^2/\lambda^2)}{(k/\lambda^2) + J(k^2/\lambda^2)}$$

$$= \frac{J+1}{12} \frac{2(2J+1) + 3J(J+1)k}{kJ + 1} .$$

The final result for the variance is then

$$Var[Y_J] = E[Y_J^2] - \{E[Y_J]\}^2$$

$$= \frac{J-1}{12} \frac{J+1}{kJ+1} .$$

Note that for the Poisson process (for which k=1.0), $Var[Y_J] = \frac{J-1}{12}$, which is also the result for the sum of J-1 uniform random variables. When k≠1.0 then, approximately,

$$Var[Y_J] \doteq \frac{J-1}{12} C^2(X),$$

where $C^2(X)$ is the coefficient of variation (which is 1/k for the gamma random variable.) Thus, the greater variability in the inter-event times is reflected directly in the increased variability of the statistic Y_J. From the point of view of a trend test, the variability of Y_J is increased to allow for the possible confusion of an actual trend with a sequence of long intervals in the stationary process. As mentioned previously, this indicates that more powerful tests for trend could be developed which use more information about the interval distribution than just the coefficient of variation.

The succeeding sections discuss determination of the

distribution of Y_J for small samples through digital computer simulation. Values of the predicted mean and variance of Y_J were calculated for each simulation run and the results are tabulated as follows (note that the mean does not depend on k):

J	MEAN	VARIANCE				
		k = .10	k = .25	k = .50	k = .75	k = 1.0
10	5.5	4.125	2.375	1.375	.971	.750
30	15.5	18.729	8.814	4.682	3.188	2.417
50	25.5	34.708	15.428	8.010	5.409	4.083
100	50.5	75.750	32.048	16.338	10.964	8.250

III. GENERATING GAMMA VARIATES FOR SIMULATION

Although there were several known methods of generating gamma random variates (see Section III.E below), all were judged deficient for large-scale computer simulation use; some are statistically inadequate for large samples and all require too much computer time. A new method was therefore developed to overcome these difficulties.

A. DESCRIPTION OF ALGORITHM

Marsaglia's rectangle-wedge-tail methods for generating normal [14,23] and exponential [22] random variates were modified to produce unit gamma variates with shape parameter k less than one. The basic approach is to select with respective probabilities $P_1, P_2, \ldots, P_n$ one of a series of random variables with distribution functions $F_1(x), F_2(x), \ldots, F_n(x)$. If x is the generated random number then

$$F(x) = P_1 F_1(x) + P_2 F_2(x) + \ldots + P_n F_n(x).$$

Taking derivatives,

$$f(x) = P_1 f_1(x) + P_2 f_2(x) + \ldots + P_n f_n(x).$$

The method is then equivalent to a geometric decompcsition of the density function of the variate to be generated. For efficiency, the P_i's and F_i's are chosen so that most of the time the generated variate is selected from a simple

distribution (preferably a uniform distribution).

Gamma variates with any desired scale parameter are obtained by multiplying a unit gamma variate (that is, one with scale parameter one) by the reciprocal of the desired scale parameter λ. Thus, the method developed generates only unit gamma variates, which have density function

$$f(x) = \frac{1}{\Gamma(k)} x^{k-1} e^{-x} .$$

The unit variates can then be scaled to produce the desired result.

It is difficult to apply the decomposition technique to the gamma distribution because the density function has a singularity at the origin when the shape parameter is less than one. A low value of x (called x_1) is thus selected as the starting point for the decomposition; values of the gamma variate less than x_1 are generated with probability P_0 by means of a series approximation to the inverse of the incomplete gamma function (see Subsection III.B.1.) A series of x_i's is then determined by the relations

$$\begin{aligned} h_1 &= .25\, x_1 \\ h_i &= 2\, h_{i-1} \\ x_i &= x_{i-1} + h_{i-1} \qquad (i=2,3,\ldots,I). \end{aligned} \tag{3-1}$$

This procedure continues until the area under the density function curve from zero to x_I is greater than 0.999; the value N is then defined to be I-1. Dependent upon k, the parameter N varies between 20 and 100. Values of the

variate greater than x_{N+1} are generated by an iterative procedure (Subsection III.B.2) with probability p_{N+1}.

The density function has thus been decomposed into N+2 regions: a "head" region below x_1, a "tail" region above x_{N+1} and a series of N vertical strips. Each strip is divided into a large rectangle and a small wedge, as shown in Figure 1. The probability for the strip as a whole can

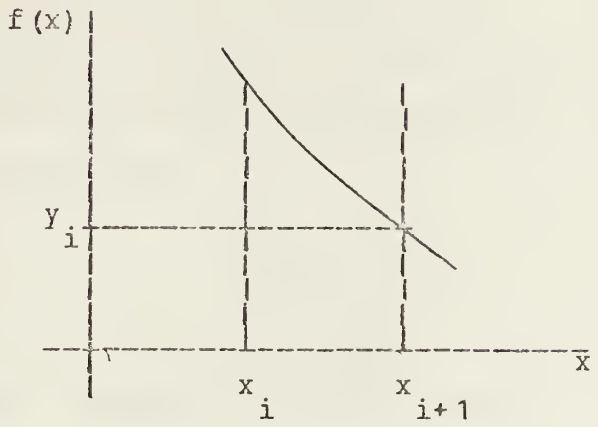

Figure 1. Decomposition of strip i.

be found by using the incomplete gamma function,

$$g(k;x) = \frac{1}{\Gamma(k)} \int_0^x y^{k-1} e^{-y}\, dy,$$

which is readily evaluated by an infinite series [1]

$$g(k;x) = \frac{1}{\Gamma(k)} \sum_{n=0}^{\infty} \frac{(-1)^n x^{n+k}}{n!\,(n+k)}$$

suitably truncated for computational purposes.

If P_i is the probability for the strip as a whole, p_i the probability of the rectangle and q_i the probability of

the wedge then the following relations exist:

$$P_i = g(k;x_{i+1}) - g(k;x_i),$$

$$p_i = h_i f(x_{i+1}),$$

$$q_i = P_i - p_i.$$

For small values of i, $q_i >> p_i$ because of the extreme steepness of f(x) near the origin; this accounts for the increasing values of h_i in (3-1). For i greater than 5, this relationship reverses so that most of the area under the f(x) curve lies within the rectangles. In fact, it was found empirically that

$$\sum_{i=1}^{N} p_i \doteq 0.72$$

for most values of k in the range 0.1 to 1.0. Thus, the gamma random variable can be sampled from a uniform distribution over 70 per cent of the time. Presumably, an increase in this probability would result from choosing each h_i optimally, but the gains to be made are small.

1. Binary Search Scheme

With the decomposition complete, there remain two problems: efficient selection of one of some 100 sampling distributions and methods for sampling from each of the four different types of distribution (head, tail, rectangle and wedge).

To formalize the selection problem, suppose that events $f_1, f_2, \ldots, f_M$ are to be selected with probabilities

$P_1, P_2, \ldots, P_M$, respectively. In the gamma generator, M = 2N+2. A cumulative probability vector $P = (P_1, P_2, \ldots, P_M)$ is defined by

$$P_i = \sum_{j=1}^{i} p_j \qquad (i=1,2,\ldots,M).$$

If a random number U uniformly distributed on (0,1) is generated, the selection of an event can be carried out by comparing U serially with $P_1, P_2, \ldots$ and stopping when, for the first time, $U \leq P_n$ for some n; event n is then selected. This method requires on the order of M/2 comparisons and would be far too slow to solve the selection problem.

Substantial saving of comparisons results from applying the binary search method to the problem of finding n such that $P_{n-1} < U \leq P_n$. For this scheme, an interval of uncertainty is defined by selection of the indices i and j so that $P_i \leq U \leq P_j$ at all times; initially, i is taken to be zero (with $P_0 = 0$) and j to be M. U is then compared with $P_{[(i+j)/2]}$ and the interval of uncertainty adjusted according to the result. This procedure is continued until the difference i-j is equal to 1; then n = j. It is not difficult to show that $\log_2 M$ comparisons are needed to find n, so that this method is considerably faster than the sequential search described previously.

A further saving of comparisons is still possible, however. Since the exact distribution of the p_i's is known,

the search procedure can be "skewed" so that more probable events can be found in fewer comparisons, thus minimizing the average number of comparisons. This can be done by comparing U with the value of P_k which is closest to the average of P_i and P_j and proceeding as in binary search. Finding the value of k for given i and j is no longer as simple, however; in the actual method used, two arrays of indices called Last and Next are precalculated. If U is less than P_k the following comparison is made with $P_{Last(k)}$; if U is greater than P_k then $P_{Next(k)}$ is used for the succeeding comparison. Termination of the search is indicated when Last(k) = 0 (choose event k) or Next(k) = 0 (choose event k+1).

The binary search scheme used here is closely related to Huffman (minimum length) codes in information theory and is essentially similar to Knuth's optimum binary search tree algorithm [15]. It can be shown [15] that between H(P) and H(P)+2 comparisons are required for this method, where H(P) is Shannon's entropy function

$$H(P) = \sum_{i=1}^{M} p_i \log_2 (1/p_i).$$

For example, for k=0.1 it was found that M = 2N + 2 = 164 and that less than 6.4 comparisons were needed on the average to select an event.

2. Rectangle Method

If the binary search scheme selects rectangle i and if V is a second uniform (0,1) random variable independent of U, then

$$Z = x_i + Vh_i$$

is a random number selected from the i-th rectangle.

3. Rejection Method for Wedges

Knuth [14] and Marsaglia [23] discuss in some detail a method for sampling a random variable whose density function is almost linear. This method is used without modification to sample from the wedges.

The values of a_i and b_i were found for each wedge so that the gamma density function $f(x)$ lies within the strip indicated in Figure 2, that is so that

$$a_i - b_i(x - x_i)/h_i \le f(x) \le b_i - b_i(x - x_i)/h_i$$

$$\text{for } x_i \le x \le x_{i+1}.$$

Since $f(x)$ is concave upwards for all x and since the

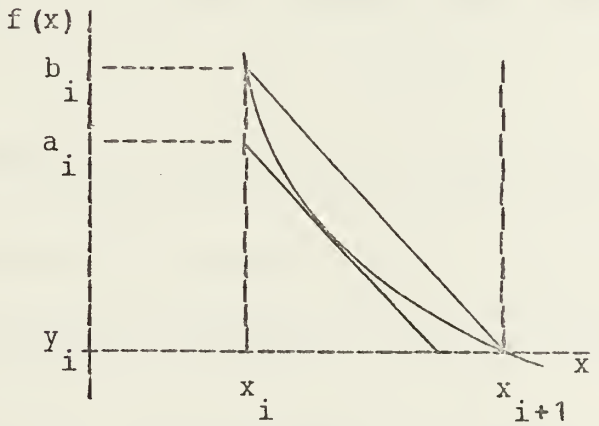

Figure 2. Sampling from the i-th wedge.

algorithm is fastest for a_i close to b_i, the lower line is taken to be the tangent to the density function curve

parallel to the chord from (x_{i+1}, y_i) to (x_i, b_i). The tangent can be found by Newton-Raphson iteration.

The wedge algorithm is then:

(1) Generate two uniform random numbers, U and V; if U > V then exchange U and V.

(2) Set Z to $x_i + Uh_i$.

(3) If $V \leq r_i = (a_i - y_i)/(b_i - y_i)$ go to step (5).

(4) If $V > U + f(Z)/b_i$ go back to step (1).

(5) Z is the desired variate; stop.

When a_i and b_i are close together then r_i will be nearly one and the algorithm will terminate as a result of step (3) most of the time. For most of the wedges in the gamma generator, values of r_i are greater than .85 so that f(x) needs to be evaluated (step (4)) less than 1/3 of the time.

Knuth [14] gives a proof that this method works properly.

B. APPROXIMATIONS FOR THE INVERSE GAMMA CDF

Any continuous random variable can be sampled if its inverse distribution function F^{-1} can be found, for if U is a uniform (0,1) variate then $Z = F^{-1}(U)$ is a variate from the desired distribution. Unfortunately, there is no simple inverse for the gamma distribution function with shape parameter less than one; Phillips and Beightler [26] attempted a rational approximation for the inverse with

little success (see Subsection III.E.2). In the present method, two different inverse approximations are used for the head and tail regions. No inverse is needed elsewhere since the wedge sampling technique described above uses $f(\cdot)$, the gamma density function, and not the gamma inverse.

1. Low Values of Z

a. Method

The series expression for the incomplete gamma function (gamma CDF) can be expanded as follows:

$$F(x) = g(k;x)$$

$$F(x) = \frac{1}{\Gamma(k)} \sum_{n=0}^{\infty} \frac{(-1)^n x^{n+k}}{n!(n+k)} \tag{3-2}$$

$$= \frac{x^k}{\Gamma(k)} \left[\frac{1}{k} - \frac{x}{k+1} + \frac{x^2}{2(k+2)} - \ldots \right].$$

If $Z = F(x)$ then

$$k\Gamma(k)Z = x^k \left[1 - \frac{k}{k+1} x + \frac{k}{2(k+2)} x^2 - \ldots \right]$$

$$[\Gamma(k+1)Z]^{1/k} = x \left[1 - \frac{k}{k+1} x + \frac{k}{2(k+2)} x^2 - \ldots \right]^{1/k}.$$

When $\frac{k}{k+1} x \ll 1$ the first-order approximation to the inverse, x_1, can be used. This corresponds to the case when the variable is to be sampled from that part of the head region close to zero.

$$x_1 = [\Gamma(k+1)Z]^{1/k}$$

$$\doteq F^{-1}(Z)$$

When the $\frac{k}{k+1}\,x$ term cannot be ignored, a better approximation is obtained by solving

$$x_1 = x[1 - \frac{k}{k+1}x + O(x^2)]^{1/k}$$

$$\doteq x[1 - \frac{1}{k} \cdot \frac{k}{k+1}\,x + O(x^2)]$$

$$\doteq x - \frac{1}{k+1}\,x^2 + O(x^3)$$

for the second-order approximation, x_2. The result is

$$x_2 = (1/2)[(k+1) - \sqrt{(k+1)^2 - 4(k+1)x_1}\,]$$

$$\doteq F^{-1}(Z).$$

A more convenient form of x_2 for computation is

$$x_2 = (2x_1) \,/\, [1 + \sqrt{1 - 4x_1/(k+1)}\,].$$

b. Accuracy

The value of P_0 (probability of the head region) is determined empirically for the gamma generator by the formula

$$P_0 = 10^{-5} \,/\, k^2$$

so that the accuracy of the approximation to the inverse incomplete gamma function was investigated for values of Z in the range zero to P_0 for various values of k. It was found that the approximation improved as Z --> 0 and as

k --> 0; in no case did the relative error exceed 10^{-9} in double precision or 10^{-7} in single precision.

2. Values of Z Near One

a. Method

Newton-Raphson iteration is used for approximating the inverse gamma distribution function for values of Z near one; that is the equation

$$h(x) = g(k;x) - Z = 0$$

is solved for x by using Newton's formula

$$x_{n+1} = x_n - h(x_n)/h'(x_n).$$

Six-digit convergence can be acheived after four or five iterations of this method starting with $x_1 = 1.0$.

Application of the method requires evaluation of h(x) and h'(x). The latter is just the gamma density function

$$h'(x) = f(x) = \frac{1}{\Gamma(k)} x^{k-1} e^{-x},$$

while the former can be found by summing the series (3-2). Since x is likely to be very large when Z is near one, however, as many as 30 terms of the series must be summed for convergence with a resultant loss of speed and the accumulation of serious round-off errors. For this reason, g(k;x) is evaluated by a continued fraction approximation [1] which has the added advantage of producing the value of 1 - g(k;x). Since the method is applied only to values of Z

in the range (0.999,1.000) the ability to deal with 1-Z leads directly to a net gain in significant digits.

b. Accuracy

It was difficult to investigate the accuracy of this approximation in the same way as for the previous method because of the loss in significant digits in 1 - Z as Z --> 1. A series of values of y in the range .00001 to .00100 was defined; for each value the approximation was used to find $x = F^{-1}(1-y)$ and then z = F(x) was obtained by the continued fraction approximation. The relative error between z and 1-y was in most cases zero and in no case greater than $4 \cdot 10^{-7}$.

C. DESCRIPTION OF COMPUTER PROGRAM

1. Program Structure

The gamma generator was implemented as a FORTRAN-callable subroutine with each call returning a single floating point (type REAL) gamma variate. One version was written in the IBM implementation of FORTRAN IV while another was written in a combination of FORTRAN and IBM 360/Assembler F. Copies of listings for both programs are on pages 77-84 and 85-88.

The basic approach in both generators is to divide the subroutine into two parts: a relatively lengthy (and slow) initialization section which calculates tables and constants and a short, fast section which is called repeatedly to handle the actual generation process. The initialization part of the program was given the name GMINIT (for GaMma INITialization) while the generator was called

GAMA. Calling sequences and arguments are discussed in the following Subsection.

GMINIT accepts values of k in the range $0.05 \leq k < 1.00$. It produces the cumulative vector PROB (analogous to P in Subsection III.A.1); before being placed into PROB the component probabilities are ordered with the smallest first to minimize round-off errors. GMINIT also finds the arrays NEXT and LAST for use in the binary search method as well as the arrays X,H,R and B for sampling from the rectangles and wedges. Finally, GMINIT calculates constants for the head and tail approximations. Sample output from GMINIT is on pages 70-72.

Two further subroutines were also written: IGAM(K,X) which evaluates the incomplete gamma function g(k;x) and INVGAM(K,Z) which uses Newton-Raphson iteration to solve the equation g(k;x) = 1-Z for x. INVGAM was also programmed in both FORTRAN and Assembly language. Listings for both IGAM and INVGAM are on pages 89-90 and 91-95, respectively.

In addition, a uniform random number generator is required. Since an average of about 2.5 uniform numbers are required for each gamma variate, timing characteristics of the uniform generator are a significant factor in the performance of the gamma generator. The particular generator used is called RANDOM and was developed by Lewis, Goodman and Miller [19]. RANDOM is very fast; it is capable of generating a uniform deviate in about 15 microseconds.

The generating routine, GAMA, carries out binary search with a single uniform random number and then samples from the proper sub-distribution using one or more additional independent uniform variates. It also performs scaling on the generated variate by multiplying it by a scale factor BETA which is an input parameter to GMINIT.

For small values of k (less than 0.1) there is some probability of a fixed point underflow error, i.e. that the generated variate will be less than the smallest possible

floating point number (about 10^{-78} for the IBM 360). The FORTRAN standard fixup for this error is to return a result of zero, which might be satisfactory for some applications; it could create serious problems, however, if, for example, a log transformation of the gamma variates is required. If a large enough scale factor is used, modification of GAMA to perform prescaling before applying the low value approximation would reduce the problem.

2. Implementations

a. FORTRAN IV Version

In the FORTRAN version of the generator the instruction sequence

```
CALL GMINIT(K,BETA)
...
CALL GAMA(Z,IX)
```

results in the initialization of the generator for shape parameter K and scale parameter $\lambda = 1/BETA$ and the assignment of a gamma random value to Z. IX is an integer seed for the random number generator RANDOM and should not normally be modified in the main program. The same sequence of gamma deviates can be reproduced at any time, however, by re-initializing IX to its original value.

IBM FORTRAN IV has several features that are not in American National Standard FORTRAN and which may not be implemented in other compilers. Two of these are used in the generator and its subroutines: the built-in function GAMMA(X) which computes the gamma function of its argument and the ENTRY statement which allows a subroutine to have multiple entry points. The ENTRY option is used to cut down on overhead in differentiating which of the two parts (GMINIT or GAMA) of the subroutine is being called; it is

equivalent to having two independent subroutines sharing a COMMON block. Besides modification of these areas, conversion to another computer would require adjustment of convergence test values in INVGAM, IGAM and GMINIT.

b. Assembler Version

The Assembler version of the generator was written to save execution time in GAMA; the instruction sequence

```
CALL START(K,BETA,IX)
...
Z = GAMA(0)
```

has the same effect as the previous sequence. Linkage overhead in GAMA has been substantially reduced, however, since no arguments need be passed (the zero in the instruction above is a dummy argument and is required by FORTRAN conventions). The capability of re-initializing the seed for RANDOM was not retained in this version.

START is a dummy section of GAMA which calls on a modified version of GMINIT. Since execution time was a secondary consideration for the initialization routine (which was normally called only once), GMINIT was maintained in FORTRAN for ease of programming. The only modifications to GMINIT in this case are in the calling sequence and in the values produced for the arrays NEXT and LAST, which were changed to allow for easier search.

3. Timing and Core Requirements

The generator was written with the sole aim of achieving the fastest possible execution time without regard for storage requirements. The execution times obtained were highly dependent on k; for example, more rectangles are needed for lower values of k so that the binary search method occupies more time in these cases. Samples of 10,000

variates were run on the Naval Postgraduate School IBM 360/67 computer and the generation times recorded for several different values of k; the average times per variate (in microseconds) were as follows:

k	FORTRAN IV Version		Assembler Version	Johnk's Method
	G Compiler	H Compiler		
0.10	430	327	201	824
0.25	385	310	189	864
0.50	350	279	186	889
0.75	339	273	178	875
0.90	325	270	179	803
Average	366	291	187	851

By way of comparison, an implementation of Marsaglia's normal generator on the same computer requires about 60 microseconds per variate while a Marsaglia exponential generator takes 70 microseconds per variate. Johnk's method is discussed in Subsection III.E.3; it is included here for purposes of comparison.

Although this method of evaluating run times is easy to implement and gives a realistic estimate of what can be expected in practice, it is somewhat misleading since competition for system resources between jobs in a multiprogrammed computer causes considerable variability in observed run times for a given job. The observations above were recorded when the system was relatively idle; on a busy afternoon these times may increase by as much as 15 per cent.

The times to complete the set-up phase in GMINIT were also highly dependent on k; the following run time values were observed:

k	Set-up Time (seconds)
0.10	.893
0.25	.348
0.50	.212
0.75	.160

Since a much larger number of rectangles is required for decomposition when k is small, it can be seen that the additional computation time for set-up in these cases can be quite costly. In fact, enough time is used to generate between 900 and 4500 variates, depending upon the value of k.

Both the Assembler and FORTRAN versions of the generator require excessive core storage, at least in comparison with other methods for generating random variates (see, for example, Ahrens and Dieter [2]). At the Naval Postgraduate School installation, where the minimum job core allotment is 100,000 bytes (25,000 words), the size of the program was not a restriction, although it could be at another installation. Both versions of the subroutine require, in addition to space for their own code and the tables, core storage for the FORTRAN built-in functions EXP, DEXP, ALOG, DLOG, GAMMA, DGAMMA and SQRT, as well as the user subroutines IGAM, INVGAM and RANDOM. Core requirements (in bytes) for the various subroutines are as follows:

	FORTRAN IV Version		Assembler
	G Compiler	H Compiler	Version
GAMA	13206	12400	5664
GMINIT			6860
INVGAM	1070	866	684
IGAM	740	574	740
RANDOM	320	320	320
Functions	5056	5056	5056
Total	20392	19216	19324

Further improvements in both space requirements and run time may still be possible through the use of in-line coding instead of subroutine calls. These modifications would have the further result of making the generator independent of vendor supplied mathematical subroutines. Ahrens and Dieter [2] point out some advantages of this approach in their improvement of Marsaglia's normal generator.

D. STATISTICAL TESTING

Having thoroughly investigated the numerical accuracy of the various approximations used in the generator, it was then necessary to test the generated variates for "statistical accuracy". Formally, the null hypothesis

H_0: The generated deviates $z_1, z_2, \ldots$ were sampled from a unit gamma distribution with shape parameter k.

was to be tested against the alternative

H_1: The generated deviates were not sampled from the unit gamma distribution.

Of the wide variety of standard goodness-of-fit tests, two were chosen for testing H_0: the chi-square test and the Anderson-Darling test. In addition, the reproductive property of the gamma distribution (that is, that the sum of two gamma random variables with shape parameters k_1 and k_2 and equal scale parameters is also a gamma random variable with shape parameter k_1+k_2) was exploited in a test on the sums of several variates.

1. Chi-square Test

A modification of the chi-square test suggested by Naylor [25] for uniform random number generators was carried out. For this test, a set of n quantiles of the object distribution (i.e., the gamma distribution) is first found;

the i-th quantile r_i is the solution to the equation

$$F(r_i) = i / n .$$

Quantiles for the gamma distribution are unique. A sample of size N variates is then generated and the number of variates f_i falling into each interval $r_{i-1} \leq z \leq r_i$ is recorded. The statistic

$$X_1^2 = \frac{n}{N} \sum_{i=1}^{n} (f_i - \frac{N}{n})^2$$

is then found for the sample; X_1^2 has approximately a chi-square distribution with n-1 degrees of freedom. This procedure is then repeated M times and a second statistic X_2^2 is calculated in an analogous manner, using m quantiles of the chi-square distribution with n-1 degrees of freedom. The value of X_2^2 is then used to accept or reject H_0.

Values chosen for the actual test were m = n = 10, N = 1000 and M = 100. Thus, each test involved generation of 100,000 variates. Tabulated results for various values of k are as follows:

k	X_2^2	Chi-square Level	Minimum X_1^2	Maximum X_1^2
.06	4.8	.149	2.24	27.28
.10	4.6	.132	1.80	22.20
.15	11.2	.738	1.90	23.74
.50	4.4	.117	1.66	21.58
.90	11.4	.751	1.66	25.68
.95	8.8	.544	1.90	21.98

The column headed "Chi-square Level" gives the quantile of the chi-square distribution with nine degrees of freedom at the observed value of X_2^2. Based on these results, H_0 could not be rejected.

There are some serious drawbacks to the chi-square test, not the least of which is the fact that the statistic depends strongly on the specific quantiles taken and is not an invariant for a particular sample. Nevertheless, the test proved to be quite sensitive even to minor variations in the generator. For example, it readily detected the statistical effect of the omission of a single wedge (whose probability was .002) by the binary search procedure, producing composite scores greater than 50 in this case. As is indicated in the sequel (Subsection III.E.2), the test was also used to reject the Phillips generator, which acheived scores greater than 800 for small values of k.

2. Anderson-Darling Test

The test of goodness-of-fit proposed by Anderson and Darling [3] is a distribution-free test in that instead of the observed values of the variate $x_1, x_2, \ldots, x_n$ the values

$$u_i = F(x_i)$$

are used. If F(x) is in fact the distribution function of the x_i's (that is, if the null hypothesis is true), then the u_i's are a sample of size n from a uniform (0,1) distribution. If the u_i's are now ordered so that $u_{(1)} \leq$

$u_{(2)} \leq \ldots \leq u_{(n)}$, then the Anderson-Darling statistic

$$W_n^2 = -n - \frac{1}{n} \sum_{i=1}^{n} [\, (2i - 1) \ln u_{(i)} + \{2(n-i) + 1\} \ln (1 - u_{(i)}) \,]$$

can be computed. The null hypothesis can then be accepted if W_n^2 is not too large. Lewis [17] has determined the distribution of W_n^2 for various values of n.

It should be mentioned that the Anderson-Darling statistic gives the most weight to observations from the tails of the distribution, where $u_{(i)}$ is very close to zero or one. Thus, the test was ideal for the gamma generator, since the rectangle-wedge technique had been successfully applied in a wide number of other cases and the main uncertainty for the generator lay in the statistical properties of the head and tail approximations.

Accordingly, Anderson-Darling statistics were found for 100 samples of 100 variates each. The results of Lewis [17] were then used to find ten quantiles of the distribution of W_{100}^2 so that a chi-square statistic could be computed for the sample of 100 Anderson-Darling statistics. The results were as follows:

k	Chi-square Statistic	Chi-square Level	Minimum W_n^2	Maximum W_n^2
.10	4.269	.107	.178	3.500
.50	4.895	.157	.175	5.071
.90	4.306	.110	.223	5.582

Once again it was decided to accept H_0 on the basis of the test.

3. "Reproductive" Property Test

It is well-known (Cramer [8]) that the sum of two independent gamma random variables with identical scale parameters and respective shape parameters k_1 and k_2 also has a gamma distribution with shape parameter k_1+k_2. Since a gamma random variable with k=1.0 is an exponential variable, a possible test for H_0 is to investigate whether the statistic

$$V_n = \sum_{i=1}^{n} z_i,$$

has a unit exponential distribution, where the z_i's are generated by GAMA with k = 1/n.

The test was carried out for samples of 100 V_n's with n = 2,4,6,8,10. For each of four such samples, an Anderson-Darling statistic was computed and the maximum value of the statistic recorded. The approximate significance level of the statistic was determined by linear interpolation in the table of the asymptotic distribution of W_n^2 given by Lewis [17]. The results are as follows:

n	k	Maximum W^2_{100}	Approximate Significance Level
2	.500	1.577	.841
4	.250	2.045	.913
6	.167	1.031	.658
8	.125	1.469	.816
10	.100	1.525	.829

As a further test of whether the statistics V_n have an exponential distribution, the statistic

$$V'_n = \text{Minimum}(V_{n;1}, V_{n;2}, \ldots, V_{n;1000})$$

was investigated; this was done specifically to test the statistical effect of the low value approximation to $F^{-1}(Z)$. It is well-known that the minimum of m unit exponential random variables is an exponential random variable with mean 1/m. Thus, samples of 100 observations of V'_n were generated for n=2,4,8 and an Anderson-Darling statistic computed using the exponential distribution function with mean .001. The results are as follows:

n	k	W^2_{100}	Approximate Significance Level
2	0.500	.903	.588
4	0.250	.431	.182
8	0.125	2.287	.936

As can be seen, H_0 is readily accepted based on these tests.

E. COMPARISON WITH OTHER METHODS

1. Naylor's Method

The method of Naylor [25] for producing gamma variates with non-integral shape parameter was not seriously considered for the simulation since it is not defined for $k < 1.0$. The method involves sampling a mixture of two gamma random variables with integral shape parameters $\lfloor k \rfloor$ and $\lfloor k \rfloor + 1$ (the notation $\lfloor k \rfloor$ denotes the truncated integer part of k). Berman has shown [28] that the method is satisfactory for small simulations when $k \geq 5$ but his results demonstrate that the method is not statistically acceptable for sample sizes on the order of 100,000 for any value of k.

The timing results for the Naylor method obtained by Berman [28] for an IBM 360/65 computer suggest that a better way to generate non-integral gamma random numbers is to add an integral gamma variate with shape parameter $\lfloor k \rfloor$ and a variate from GAMA with shape parameter $k - \lfloor k \rfloor$. For $k > 10$ this modification adds less than 20 per cent to the execution time but renders the result statistically exact.

2. Phillips's Method

Phillips and Beightler [26] proposed the use of a rational approximation to the inverse gamma distribution function for generating gamma variates; three different approximations were developed for $0 < k \leq 2.0$, $2.0 < k \leq 5.0$ and $k > 5.0$. The technique was extensively tested by Phillips using the Kolmogorov-Smirnov goodness of fit test; unfortunately, this test is notoriously insensitive to deviations in the tails of the distribution. The chi-square goodness of fit test was therefore applied to the method for

$k < 1.0$ and it was found that the hypothesis that the Phillips numbers were sampled from the gamma distribution could be rejected at any desired confidence level (chi-square scores in excess of 800 on 9 degrees of freedom were observed). The method was not investigated for $k>1.0$. Furthermore, this method required substantially more execution time than the method developed here (on the order of 600 microseconds per variate).

3. Johnk's Method

Johnk's method [13] was the only known theoretically exact means of generating gamma variates with non-integral shape parameter. It is a rejection technique, like the wedge method of of Subsection III.A.3. For generating gamma variates with shape parameter k less than 1.0 the method can be set forth as follows:

(1) Generate two independent uniform (0,1) random variates U and V.

(2) Set $X = U^{1/k}$ and $Y = V^{1/(1-k)}$.

(3) If $X + Y > 1.0$ go back to (1).

(4) Generate a new uniform variate R and set $W = \ln R$.

(5) $Z = W \cdot X / (X+Y)$ is a unit gamma variate. Scale Z if required and stop.

Translations of Johnk's original proofs of the correctness of this methcd are contained in the Phillips and Beightler [26] and Fox [9] papers. Essentially the method generates a beta variate and then uses the Lukacs-Laha results [16,21] to produce a gamma variate. Johnk shows that when k is less than 1.0 the rejection in step (3) will occur between 1.00 and 1.27 times per variate on the average.

Johnk's method is not satisfactory for large-scale simulation use because it requires excessive time; see

Subsection III.C.3 for the observed times for a FORTRAN implementation of the method. The time can be improved by recognizing that if U is a uniform (0,1) variate then -ln U is a unit exponential variate. The time to generate U and evaluate its natural logarithm is about 130 microseconds on the IBM 360/67 computer while an assembler implementation of Marsaglia's exponential algorithm can deliver an exponential variate in about 70 microseconds. The improved algorithm is then

(1) Generate two independent exponential variates, R and S.

(2) Set $X = e^{-R/k}$ and $Y = e^{-S/(1-k)}$.

(3) If $X + Y > 1$ go back to (1).

(4) Generate a new exponential variate T.

(5) Deliver $Z = T \cdot X/(X+Y)$. Stop.

Even with these improvements, however, the method still requires over 500 microseconds per variate in an assembler version. Thus, allowing for GAMA's substantial set-up time, it is faster to use the Johnk method for less than 250 variates (k=0.75) to 1,350 variates (k=0.1), depending on k, but the method is too slow for large-scale simulation (100,000 or more variates). The method remains very attractive for use when time is not so critical, however, since it is very easy to program and requires minimal storage.

IV. COMPUTER SIMULATION EXPERIMENT

A. DESCRIPTION OF METHOD

The basic purpose of the simulation experiment was to investigate the distribution of Y_J for small values of J. Different simulations were run for shape parameters k of .1, .25, .5 and .75 corresponding to coefficients of variation $C^2(X)$ of 10, 4, 2 and 1.33. In addition, a simulation using an implementation of the Marsaglia exponential generator was run for the case k=1.0.

1. Simulation

Ten thousand samples of J gamma variates each were generated for J = 10,30,50,100 and the Y_J statistic computed for every sample. The first four sample moments were found and then the entire sample of 10,000 Y_J's was ordered in order to estimate the quantiles of the distribution (see Subsection IV.B.2). This procedure produced 27 statistics: four sample moment estimates and 23 quantile estimates. The entire procedure was then repeated ten times and the mean and variance of each of the 27 estimates found over the ten samples of each.

Obviously a great many gamma variates were required to carry out the simulation, nearly 20 million for each value of k. The size of the simulation graphically demonstrates the need for speed in the gamma generator.

Even with the relatively fast performance of the Assembly-coded generator, the total IBM 360/67 central processor time needed to run the simulation was about 60 minutes for each k value.

2. FORTRAN IV Program

The simulation itself was written in FORTRAN IV; a program listing is included on pages 96-98. The program and subroutines required nearly 89,000 bytes of core at run time, 40,000 bytes of which were needed for the 10,000-element array used for quantile estimation. A sample of output from the program for J=30 is included on pages 73-76.

B. ESTIMATION OF PARAMETERS

1. Distribution Moments

If $Y_{J;i}$ represents the i-th observation of Y_J in the simulation, then the r-th moment m_r of Y_J can be estimated by

$$\bar{m}_r = \frac{1}{N} \sum_{i=1}^{N} (Y_{J;i})^r .$$

An unbiased estimator of the variance is then

$$\bar{s}^2 = \frac{N}{N-1} (\bar{m}_2 - \bar{m}_1^2) .$$

In the simulation, estimates of the first four moments (r=1,2,3,4) were obtained. In addition, the sample variance and the coefficients of skewness and kurtosis were

calculated from the moment estimates.

2. Quantiles

a. Quantile Estimation Considerations

A quantile x_a of a distribution $F(x)$ is defined as the solution to the equation

$$F(x_a) = a \qquad (0 < a < 1).$$

Quantiles are unique for most continuous distributions but are not, in general, for discrete distributions. In the simulation, quantiles of Y_J were estimated for a = .001, .002, .005, .010, .020, .025, .050, .1(.1).9, .950, .975, .980, .990, .995, .998 and .999.

There are two principal methods for quantile estimation in simulations: the order statistic estimator and the Robbins-Monro stochastic approximation technique. (For a more complete discussion, see Goodman, Lewis and Robbins [11].) The order statistic method was chosen for this simulation because of the ease of implementation. Let the N observations $Y_{J;1}, Y_{J;2}, \ldots, Y_{J;N}$ of Y_J be ordered so that $Y_{(J;1)} \leq Y_{(J;2)} \leq \ldots \leq Y_{(J;N)}$; then $Y_{(J;i)}$ is called the i-th order statistic of the sample and an estimate of the a-quantile y_a of the distribution of Y_J is

$$\bar{y}_a = Y_{(J;\lfloor aN \rfloor)}.$$

b. Sorting Random Data

In order to estimate the extreme quantiles of the distribution of Y_J it is necessary to deal with large samples of Y_J. Considerations of bias (see the next Subsection) also call for large samples. As mentioned previously, a sample size of 10,000 was chosen for quantile estimation.

Sorting such a large sample can introduce considerable overhead into the simulation, however, unless the sorting method is carefully chosen. There is a bewildering variety of known sorting methods for various purposes (Knuth [15] devotes an entire text to the subject). The particular technique selected for the simulation is due to Singleton [27]; a FORTRAN implementation of the method is capable of ordering a sample of 10,000 normalized type REAL numbers in less than four seconds. In the simulation, therefore, less than five per cent of the execution time was spent in sorting the samples.

3. Errors and Bias

The parameter estimates may not correspond to the true parameter values for three reasons: statistical variability, computation errors and estimator bias. Statistical variability is an inescapable reflection of the fact that a simulation is not a deterministic process, as is the summing of an infinite series. Thus, the probability that the precise expected value of an estimator will be observed is very low and may be zero for a continuous distribution. The law of large numbers, however, guarantees that the estimator will converge closer to its theoretical value as more replications of the simulation are performed.

Computation error is a reflection of digital

computer round-off errors; that is, for every computer system there is a floating-point constant e such that when $|d| \leq |e|$

$$1.0 + d = 1.0.$$

For the IBM 360 system, $e = 9.53 \cdot 10^{-7}$. Thus, if very many data elements are to be summed (as in determining the moment estimators) there may be some loss of significance as the accumulated sum increases. The problem will be even more acute if very skewed data are to be added, as in the determination of Y_{1J} and Y_{2J}.

Round-off errors in Y_{1J} and Y_{2J}, however, tend to cancel out when Y_J is calculated. On the other hand, in the simulation the observed moment estimators $\bar{m}_r$ were consistently below their theoretical values, thus reflecting considerable round-off error. In fact, $\bar{m}_r$ was less than m_r in 17 observations out of 20 for r=1 and in 14 of 20 for r=2. Computing the moment estimators using double precision arithmetic (thus decreasing the value of e to $2.22 \cdot 10^{-16}$ and therefore greatly reducing round-off errors) resulted in values of $\bar{m}_r$ more in agreement with theory.

Estimator bias is a theoretical property of the specific estimators used in a simulation. In this case, the $\bar{m}_r$ are unbiased estimators of the m_r but

$$E[\bar{Y}_a] = y_a + O(\tfrac{1}{N}),$$

where N is the sample size (Goodman, Lewis and Robbins [11]). Thus $\bar{y}_a$ is a biased estimator. No attempt was made to evaluate the extent of the bias of $\bar{y}_a$, but it is clear that the simulation results may not be accurate for more than two or three decimal places.

C. SIMULATION RESULTS

1. Agreement with Theory

For each simulation run, a t-statistic was computed using the sample mean and variance. Since the sample mean is the sum of 100,000 identically distributed random variables (the observations of Y_J), it is safe to assume that its distribution is very nearly normal and that

$$t = (\bar{m}_1 - E[Y_J]) \; / \; (\bar{s} \; / \; \sqrt{n})$$

where

$$\bar{s}^2 = \frac{n}{n-1} \quad (\bar{m}_2 - \bar{m}_1^2)$$

has a t distribution with n degrees of freedom. When n is large (100,000 in this case) then the t distribution can be approximated by the standard normal distribution. The t statistics for the various runs are summarized in the following table:

J	10	30	50	100
k				
0.10	-0.0688	-2.2718	-0.5641	-0.9348
0.25	1.0851	-2.7454	-2.4309	-1.9453
0.50	0.1989	-3.6603	-2.5376	1.2218
0.75	-0.1440	-1.8950	-2.6556	-1.2765
1.00	-0.5925	-3.4506	-1.2623	NA

Since the .01 quantile of the standard normal distribution is -2.326, it can be readily seen that the above results do not agree with theory.

As mentioned above, the reason for this discrepancy lies in accumulating round-off errors in calculating the moment estimators. The simulations for J=30 were thus rerun with the moment estimators computed in double precision. No change in the quantile estimates was observed, but the new t statistics were

k	0.10	0.25	0.50	0.75
t statistic	-0.9206	-0.8101	-1.0592	1.3176

Thus, it is concluded that the simulation results agree with theory as far as the observed moments are concerned.

2. Tabulation of the Distribution of Y_J

In addition to the actual values for Y_J, the simulation program produced normalized values for the quantile estimates

$$\bar{n}_a = (\bar{y}_a - u) / \sigma ,$$

where

$$u = E[Y_J]$$

$$= \frac{J+1}{2}$$

and

$$\sigma^2 = Var[Y_J]$$

$$= \frac{J-1}{12} \frac{J+1}{kJ+1} .$$

The normalized statistics are tabulated here in order to show the asymptotic convergence of the distribution of Y_J to a normal distribution; quantiles of the standard normal distribution are also listed.

Alpha	J=10	J=30	J=50	J=100	Normal Quantile
.001	-2.2027	-2.7519	-2.8819	-2.9941	-3.090
.002	-2.1920	-2.6114	-2.7360	-2.7978	-2.878
.005	-2.1473	-2.4244	-2.4940	-2.5415	-2.576
.010	-2.0788	-2.2439	-2.2911	-2.3166	-2.326
.020	-1.9445	-2.0244	-2.0507	-2.0549	-2.054
.025	-1.8738	-1.9473	-1.9655	-1.9672	-1.960
.050	-1.6546	-1.6654	-1.6675	-1.6547	-1.645
.100	-1.3444	-1.3222	-1.3107	-1.2983	-1.282
.200	-0.9230	-0.8844	-0.8697	-0.8563	-0.842
.300	-0.5885	-0.5541	-0.5441	-0.5364	-0.524
.400	-0.2773	-0.2746	-0.2608	-0.2613	-0.253
.500	-0.0012	-0.0051	0.0023	-0.0029	0
.600	0.2802	0.2687	0.2685	0.2604	0.253
.700	0.5923	0.5535	0.5535	0.5382	0.524
.800	0.9257	0.8783	0.8732	0.8566	0.842
.900	1.3426	1.3177	1.3042	1.2958	1.282
.950	1.6541	1.6553	1.6621	1.6577	1.645
.975	1.8754	1.9228	1.9537	1.9627	1.960
.980	1.9437	2.0036	2.0467	2.0523	2.054
.990	2.0777	2.2178	2.2859	2.3096	2.326
.995	2.1498	2.4068	2.5013	2.5484	2.576
.998	2.1901	2.6027	2.7520	2.8258	2.878
.999	2.2009	2.7274	2.9476	3.0085	3.090
u	5.5	15.5	25.5	50.5	
σ	2.0310	4.3277	5.8914	8.7034	
t-Test for Mean	-0.0688	-2.2718	-0.5641	-0.9348	

Table I. Simulation results for k=0.10. Normalized quantiles of the distribution of Y_J under the null hypothesis b=0 (no trend) are tabulated.

Alpha	J=10	J=30	J=50	J=100	Normal Quantile
.001	-2.6635	-3.0231	-2.9940	-3.0984	-3.090
.002	-2.5614	-2.7949	-2.8016	-2.8912	-2.878
.005	-2.3792	-2.5528	-2.5519	-2.5871	-2.576
.010	-2.2094	-2.3081	-2.3277	-2.3405	-2.326
.020	-2.0089	-2.0596	-2.0612	-2.0672	-2.054
.025	-1.9340	-1.9722	-1.9727	-1.9775	-1.960
.050	-1.6581	-1.6723	-1.6639	-1.6715	-1.645
.100	-1.3211	-1.3174	-1.2980	-1.2982	-1.282
.200	-0.8811	-0.8663	-0.8604	-0.8583	-0.842
.300	-0.5511	-0.5414	-0.5375	-0.5339	-0.524
.400	-0.2643	-0.2608	-0.2632	-0.2585	-0.253
.500	0.0089	-0.0016	-0.0070	0.0015	0.
.600	0.2673	0.2573	0.2568	0.2573	0.253
.700	0.5619	0.5365	0.5311	0.5351	0.524
.800	0.8913	0.8639	0.8589	0.8529	0.842
.900	1.3230	1.3024	1.3015	1.2946	1.282
.950	1.6566	1.6636	1.6593	1.6576	1.645
.975	1.9359	1.9642	1.9692	1.9776	1.960
.980	2.0101	2.0579	2.0545	2.0682	2.054
.990	2.2225	2.2991	2.3178	2.3388	2.326
.995	2.3901	2.5438	2.5635	2.5759	2.576
.998	2.5484	2.7867	2.8170	2.8870	2.878
.999	2.6496	2.9408	3.0409	3.0975	3.090
u	5.5	15.5	25.5	50.5	
σ	1.5353	2.9688	3.9276	5.6611	
t-Test for Mean	1.0851	-2.7454	-2.4309	-1.9453	

Table II. Simulation results for k=0.25. Normalized quantiles of the distribution of Y_J under the null hypothesis b=0 (no trend) are tabulated.

Alpha	J=10	J=30	J=50	J=100	Normal Quantile
.001	-2.8514	-3.0592	-3.0958	-3.0890	-3.090
.002	-2.7088	-2.8700	-2.9104	-2.9028	-2.878
.005	-2.4928	-2.5669	-2.6023	-2.5900	-2.576
.010	-2.2861	-2.3410	-2.3446	-2.3538	-2.326
.020	-2.0416	-2.0851	-2.0823	-2.0925	-2.054
.025	-1.9578	-1.9918	-1.9941	-1.9955	-1.960
.050	-1.6617	-1.6761	-1.6750	-1.6605	-1.645
.100	-1.3028	-1.3107	-1.3119	-1.2956	-1.282
.200	-0.8685	-0.8723	-0.8620	-0.8468	-0.842
.300	-0.5420	-0.5460	-0.5375	-0.5201	-0.524
.400	-0.2661	-0.2660	-0.2587	-0.2471	-0.253
.500	0.0030	-0.0051	0.0022	0.0092	0.
.600	0.2665	0.2592	0.2596	0.2708	0.253
.700	0.5483	0.5375	0.5380	0.5430	0.524
.800	0.8727	0.8610	0.8622	0.8679	0.842
.900	1.3098	1.3037	1.3123	1.3172	1.282
.950	1.6615	1.6712	1.6747	1.6861	1.645
.975	1.9576	1.9847	1.9851	2.0026	1.960
.980	2.0483	2.0753	2.0827	2.0992	2.054
.990	2.2750	2.3388	2.3416	2.3790	2.326
.995	2.4723	2.5856	2.5955	2.6282	2.576
.998	2.6926	2.8816	2.8901	2.9099	2.878
.999	2.8498	3.0650	3.0638	3.1490	3.090
u	5.5	15.5	25.5	50.5	
σ	1.1726	2.1639	2.8301	4.0421	
t-Test for Mean	0.1989	-3.6603	-2.5376	1.2218	

Table III. Simulation results for k=0.50. Normalized quantiles of the distribution of Y_J under the null hypothesis b=0 (no trend) are tabulated.

Alpha	J=10	J=30	J=50	J=100	Normal Quantile
.001	-2.9330	-3.1428	-3.1428	-3.1331	-3.090
.002	-2.7684	-2.9370	-2.9344	-2.9034	-2.878
.005	-2.5103	-2.6022	-2.5863	-2.6140	-2.576
.010	-2.2857	-2.3462	-2.3430	-2.3620	-2.326
.020	-2.0410	-2.0911	-2.0684	-2.0876	-2.054
.025	-1.9544	-1.9931	-1.9815	-1.9907	-1.960
.050	-1.6617	-1.6686	-1.6722	-1.6678	-1.645
.100	-1.3090	-1.3054	-1.3063	-1.3039	-1.282
.200	-0.8670	-0.8556	-0.8629	-0.8497	-0.842
.300	-0.5429	-0.5275	-0.5351	-0.5279	-0.524
.400	-0.2628	-0.2496	-0.2578	-0.2495	-0.253
.500	0.0024	0.0079	0.0003	0.0064	0
.600	0.2618	0.2658	0.2603	0.2631	0.253
.700	0.5410	0.5380	0.5358	0.5390	0.524
.800	0.8707	0.8640	0.8574	0.8584	0.842
.900	1.3153	1.3052	1.3056	1.3000	1.282
.950	1.6657	1.6771	1.6803	1.6708	1.645
.675	1.9598	1.9889	1.9920	1.9943	1.960
.980	2.0509	2.0705	2.0849	2.0894	2.054
.990	2.2908	2.3475	2.3639	2.3518	2.326
.995	2.5102	2.6129	2.6022	2.5834	2.576
.998	2.7933	2.8948	2.9219	2.8904	2.878
.999	2.9584	3.1079	3.0848	3.1105	3.090
u	5.5	15.5	25.5	50.5	
σ	0.9852	1.7855	2.3257	3.3112	
t-Test for Mean	-0.1440	-1.8950	-2.6556	-1.2765	

Table IV. Simulation results for k=0.75. Normalized quantiles of the distribution of Y_J under the null hypothesis b=0 (no trend) are tabulated.

Alpha	J=10	J=30	J=50	J=100	Normal Quantile
.001	-2.9858	-3.0964	-3.0550	NA	-3.090
.002	-2.7774	-2.8721	-2.8522		-2.878
.005	-2.5046	-2.5750	-2.5525		-2.576
.010	-2.2994	-2.3304	-2.3006		-2.326
.020	-2.0442	-2.0496	-2.0350		-2.054
.025	-1.9502	-1.9566	-1.9455		-1.960
.050	-1.6517	-1.6464	-1.6489		-1.645
.100	-1.2917	-1.2846	-1.2775		-1.282
.200	-0.8543	-0.8440	-0.8348		-0.842
.300	-0.5284	-0.5244	-0.5172		-0.524
.400	-0.2540	-0.2533	-0.2409		-0.253
.500	-0.0009	-0.0004	0.0075		0
.600	0.2562	0.2566	0.2607		0.253
.700	0.5286	0.5238	0.5317		0.524
.800	0.8519	0.8422	0.8443		0.842
.900	1.2905	1.2875	1.2843		1.282
.950	1.6536	1.6417	1.6508		1.645
.975	1.9521	1.9635	1.9690		1.960
.980	2.0417	2.0588	2.0541		2.054
.990	2.2855	2.3152	2.3165		2.326
.995	2.4933	2.5520	2.5828		2.576
.998	2.7571	2.8427	2.8756		2.878
.999	2.9209	3.0438	3.0758		3.090
u	5.5	15.5	25.5		
σ	0.8660	1.5546	2.0207		
t-Test for Mean	-0.5925	-3.4506	-1.2623		

Table V. Simulation results for k=1.00. Normalized quantiles of the distribution of Y_J under the null hypothesis b=0 (no trend) are tabulated.

D. LARGE-SCALE SIMULATION TOOLS

In programming the simulation it became clear that a set of standard data-handling subroutines would have been most helpful for parameter estimation. The facilities of standard simulation languages (e.g., GPSS or CSMP) and data reduction programs (SNAP/IEDA or SPSS) are not suited to the amount or type of information generated in large-scale statistical simulations such as this one. Extensive research in the literature was thus required to find high-speed techniques for sorting, quantile estimation and statistical testing and considerable effort was expended in programming the techniques.

The sophistication of state-of-the-art methods in random number generation, for example, can result in substantial savings of computer time and core requirements but only at the expense of program complexity. Since the specific details of the method involved are usually of peripheral importance to the simulation itself, removal of the programming burden will result in more resources becoming available for the main purpose of the simulation.

Following Lewis' discussion [24] of a large-scale computer-aided statistical package, the following routines would have been of value in the Y_J simulation:

(1) A fully-documented quantile estimation subroutine using the maximum transformation Robbins-Monro stochastic approximation technique.

(2) One or more efficient goodness-of-fit test subroutines: t-test, chi-square test, Anderson-Darling test, Kolmogorov-Smirnov test.

(3) A sorting routine equal in speed to ACM Algorithm 347 [27].

(4) An efficient subroutine for accumulating moment estimators with minimal round-off error.

V. CONCLUSION

A. APPLICABILITY OF RESULTS

1. Gamma Generator

In view of the wide variety of physical events which can be successfully modeled empirically as gamma renewal processes, the availability of an exact, high-speed gamma variate generator should enable more extensive simulations to be conducted in many applications areas. If the shape parameter remains constant throughout a given application, pre-calculation of GAMA's tables and constants will save set-up time in succeeding runs; if not, then the trade-off between GAMA and the Johnk technique (Subsection III.E.3) should be considered in selecting a generator. Nevertheless, it is clear that whenever a large number of gamma variates with shape parameter less than one is required, GAMA is superior to any other known method for generating them.

2. Tests for Trend

The results of Section II.D indicate that the variance of the standard B test for the Poisson process needs to be "inflated" by the coefficient of variation squared for use in a gamma renewal process. Thus, it may now be possible to accept the null hypothesis that a given process is without trend when prior to this result it was not. For example, Lewis and Shedler [20] in analyzing computer page exception data obtained B values which were

far too high to accept the null hypothesis (normalized values of -2.83, -8.67 and -18.11 were observed; the .01 quantile is -2.33). However, allowing for the extremely high coefficients of variation involved (3.34, 3.27 and 3.70), the null hypothesis can be accepted in two of the three cases.

Since in practice the actual value of k is seldom known, the results obtained in Chapter II suggest that the estimated sample coefficient of variation could be used as a multiplier of the Poisson variance in applying the B test. The simulation results indicate that this procedure may lead to slightly higher acceptance levels in the gamma case when k is small, but the effect diminishes with increasing J.

Without investigation of the relative power of the Y_J test against other standard trend tests it may not be wise to apply it to an arbitrary process. It would appear most useful, in any case, when small values of k are involved, for this is precisely the situation which the B test does not handle well.

As for the tabular results, Tables I - V are sufficiently accurate for routine application of the test based on B, especially since the level of the test is arbitrary. Possible inaccuracies in the tabular values might be significant, though, if the values were used to investigate the power of the Y_J test against the alternatives $b \neq 0$. This has not been done.

The distributions of Y_J are in all cases observed to be approximately symmetric and underdispersed with respect to the normal distribution. Convergence to the to the normal distribution is rapid, the rate being slowest, as expected, for k=0.1. Even there, the normal approximation is adequate for J=50 if a level of no higher than 0.990 and no lower than 0.010 is required. The approximate rates of

convergence of Y_J are such that it appears possible to use the normal approximation whenever J is greater than the following values:

k	0.10	0.25	0.50	0.75	1.00
J	100	50	30	30	10

B. AREAS FOR FURTHER STUDY

1. Gamma Generator

The problem of a general purpose gamma generator ($k \geq 1$ as well as $k < 1$) should be investigated in order to extend the present work to wider applications areas. A straightforward way to do this is to follow Berman's method [28] of adding $\lfloor k \rfloor$ exponential variates to a gamma variate with shape parameter $k - \lfloor k \rfloor$. This approach has two distinct drawbacks in the present case:

(1) When $k - \lfloor k \rfloor$ is small, GAMA requires much more setup and generation time than when $0.5 \leq k - \lfloor k \rfloor < 1.0$; when $k - \lfloor k \rfloor < 0.05$, GAMA cannot be used at all. This difficulty could be overcome by using the fact that the square of a standard normal random variable has the gamma distribution with $k=0.5$, $\lambda=0.5$; thus, if Z is a standard normal variate then $Y = 2Z^2$ is a unit gamma variate with shape parameter 0.5. If a Marsaglia normal generator is available, Y can be readily generated and added to a series of exponential variates to allow GAMA to be called with shape parameter in the optimum range. A thorough investigation of the time requirements for the various options is needed to find the fastest method in any given case.

(2) When k is large, generation of $\lfloor k \rfloor$ exponential variates may require excessive time. In this case, taking the logarithm of the product of $\lfloor k \rfloor$ uniform variates may be a more efficient way to proceed.

It may be possible to overcome both these difficulties by extending the decomposition technique to values of k greater than one. In this case, however, the density function is no longer monotone and automatic generation of the sub-distributions and their probabilities will be considerably more difficult.

2. Tests for Trend

As mentioned previously, a more thorough investigation of the theoretical properties of the Y_J test is also indicated. Its relative power compared to other tests against the trend model should be determined; possible examples of such tests include a normal theory regression test for trend after a logarithmic transformation or the various non-parametric tests for trend that are essentially equivalent to Kendall's rank correlation test. Results should also be obtained for other types of renewal process.

GENERATED VALUES FOR K= 5.000000E-01 BETA= 1.000000E 00

HEAD PROBABILITY= 2.000011E-04 TAIL PROBABILITY= 4.005432E-05

RECTANGLE/WEDGE VALUES

I	X(I)	P(I)	Q(I)	R(I)	B(I)
1	3.141626E-08	2.236078E-05	1.246153E-06	9.581481E-01	3.169017E 03
2	3.927033E-08	3.779664E-05	3.171779E-06	9.369842E-01	2.819253E 03
3	5.497846E-08	6.030221E-05	6.785616E-06	9.154791E-01	2.365045E 03
4	8.639472E-08	9.176671E-05	1.246110E-05	8.979554E-01	1.872624E 03
5	1.492272E-07	1.352253E-04	2.049345E-05	8.860415E-01	1.416690E 03
6	2.748922E-07	1.954720E-04	3.145642E-05	8.789549E-01	1.039967E 03
7	5.262222E-07	2.795863E-04	4.643179E-05	8.750603E-01	7.500715E 02
8	1.028881E-06	3.976775E-04	6.711972E-05	8.730119E-01	5.358142E 02
9	2.034201E-06	5.640355E-04	9.597931E-05	8.719643E-01	3.808430E 02
10	4.044841E-06	4.988284E-04	1.364939E-04	8.714308E-01	2.699988E 02
11	8.066121E-06	1.130531E-03	1.935721E-04	8.711599E-01	1.911670E 02
12	1.610868E-05	1.599371E-03	2.741409E-04	8.710284E-01	1.352630E 02
13	3.219380E-05	2.262193E-03	3.879750E-04	8.709646E-01	9.567520E 01
14	6.436404E-05	3.199311E-03	5.489010E-04	8.709354E-01	6.766138E 01
15	1.287045E-04	4.524134E-03	7.764772E-04	8.709244E-01	4.784447E 01
16	2.573854E-04	6.396592E-03	1.098432E-03	8.709329E-01	3.382796E 01
17	5.147473E-04	9.041585E-03	1.554139E-03	8.709592E-01	2.391394E 01
18	1.029471E-03	1.277366E-02	2.199732E-03	8.710189E-01	1.690067E 01
19	2.058919E-03	1.802759E-02	3.115989E-03	8.711437E-01	1.193764E 01
20	4.117813E-03	2.539011E-02	4.421089E-03	8.713853E-01	8.422844E 00
21	8.235604E-03	3.561255E-02	6.292313E-03	8.718535E-01	5.929946E 00
22	1.647118E-02	4.954105E-02	9.008113E-03	8.727235E-01	4.156673E 00
23	3.294235E-02	6.779116E-02	1.302999E-02	8.742151E-01	2.888266E 00
24	6.588465E-02	8.975840E-02	1.914674E-02	8.763657E-01	1.971886E 00
25	1.317693E-01	1.112664E-01	2.859479E-02	8.781085E-01	1.299223E 00
26	2.635386E-01	1.208997E-01	4.238629E-02	8.752680E-01	7.963005E-01
27	5.270772E-01	1.009331E-01	5.711794E-02	8.578054E-01	4.207527E-01
28	1.054154E 00	4.974341E-02	5.672789E-02	8.082208E-01	1.638207E-01
29	2.108309E 00	8.543260E-03	2.780234E-02	7.037997E-01	3.441114E-02
30	4.216618E 00	1.781907E-04	3.466037E-03	5.373893E-01	2.197152E-03
SUMS		7.211952E-01	2.785645E-01		

VALUES FOR HEAD/TAIL APPROXIMATION:
J1= 26
H1= 0.177246E-03 H2= 0.200000E 01 H3= 0.330000E-06 H4= 0.266667E 01

STARTING POINT FOR BINARY SEARCH 63

PROBABILITY VECTOR AND LINKS

	PROB	TABLE	NEXT	LAST
7	1.246153E-06	-1	0	0
8	4.417932E-06	-2	0	7
9	1.120355E-05	-3	0	8
10	2.366464E-05	-4	0	9
11	4.415809E-05	-5	0	10
12	6.651887E-05	1	13	11
13	9.797529E-05	-6	0	0
14	1.357719E-04	2	16	12
15	1.758263E-04	0	0	0
16	2.222580E-04	-7	0	15
17	2.825602E-04	3	19	14
18	3.496800E-04	-8	0	0
19	4.414467E-04	4	0	18
20	5.374260E-04	-9	22	17
21	6.726512E-04	5	0	0
22	8.091452E-04	-10	0	21
23	9.873358E-04	30	26	20
24	1.180908E-03	-11	0	0
25	1.376380E-03	6	0	24
26	1.576381E-03	0	27	25
27	1.850522E-03	-12	0	0
28	2.130108E-03	7	31	23
29	2.518083E-03	-13	30	0
30	2.915761E-03	8	0	0
31	3.464662E-03	-14	0	29
32	4.028697E-03	9	35	28
33	4.805174E-03	-15	0	0
34	5.603999E-03	10	0	33
35	6.702431E-03	-16	0	34
36	7.832959E-03	11	39	32
37	9.387098E-03	-17	0	0
38	1.098647E-02	12	0	37
39	1.318620E-02	-18	40	38
40	1.544839E-02	13	0	0
41	1.856438E-02	-19	43	36
42	2.176369E-02	14	0	0
43	2.522972E-02	-30	44	42
44	2.965081E-02	-20	0	0
45	3.417495E-02	15	48	41
46	4.046726E-02	-21	47	0
47	4.686385E-02	16	0	0

48	5.540711E-02	29	0	46
49	6.441522E-02	-22	52	45
50	7.345676E-02	17	0	0
51	8.623040E-02	18	0	50
52	9.926039E-02	-23	53	51
53	1.172879E-01	19	0	0
54	1.364347E-01	-24	56	49
55	1.618248E-01	20	0	0
56	1.896271E-01	-29	57	55
57	2.182218E-01	-25	0	0
58	2.538344E-01	21	61	54
59	2.962207E-01	-26	0	0
60	3.457617E-01	22	0	59
61	3.955050E-01	28	62	60
62	4.522329E-01	-28	0	0
63	5.093508E-01	-27	66	58
64	5.771420E-01	23	0	0
65	6.669004E-01	24	0	64
66	7.678335E-01	27	67	65
67	8.790999E-01	25	0	0
68	1.000000E 00	26	0	0

SIMULATION RESULTS FOR K= 0.10 AND SAMPLE SIZE = 30

THE FIRST FOUR MOMENTS ARE:

	1.548733D 01	2.587971D 02	4.594324D 03	8.570594D 04
VARIANCE:	2.096324E-03	2.444954E 00	1.802261E 03	1.164885E 06

MEAN = 1.548733E 01

VARIANCE = 1.894167E 01 STANDARD DEVIATION = 4.352202E 00

SKEWNESS = -5.621020E-03

KURTOSIS = 2.626644E 00

QUANTILES OF THE DISTRIBUTION:

ALPHA	ALPHA-QUANTILE	VARIANCE	NORMALIZED QUANTILE
0.001	3.590595E 00	2.686967E-02	-2.751887
0.002	4.198548E 00	1.481914E-02	-2.611408
0.005	5.007994E 00	1.301374E-02	-2.424372
0.010	5.789165E 00	4.642975E-03	-2.243867
0.020	6.739015E 00	8.839302E-03	-2.024387
0.025	7.072540E 00	6.934214E-03	-1.947320
0.050	8.292786E 00	9.536628E-03	-1.665359
0.100	9.777983E 00	3.388013E-03	-1.322177
0.200	1.167267E 01	1.095100E-03	-0.884375
0.300	1.310211E 01	2.725954E-03	-0.554076
0.400	1.431170E 01	5.301170E-03	-0.274578
0.500	1.547803E 01	7.272765E-03	-0.005077
0.600	1.666309E 01	2.811005E-03	0.268752
0.700	1.789522E 01	1.637549E-03	0.553459
0.800	1.930099E 01	3.406955E-03	0.878290
0.900	2.120285E 01	5.729992E-03	1.317748
0.950	2.266379E 01	1.310683E-02	1.655325
0.975	2.382124E 01	1.734190E-02	1.922776
0.980	2.417113E 01	2.280266E-02	2.003624
0.990	2.509793E 01	1.732585E-02	2.217779
0.995	2.591594E 01	2.951581E-02	2.406795
0.998	2.676401E 01	6.331199E-02	2.602757
0.999	2.730328E 01	1.141205E-01	2.727366

SIMULATION RESULTS FOR K= 0.25 AND SAMPLE SIZE = 30

THE FIRST FOUR MOMENTS ARE:

	1.549231D 01	2.490223D 02	4.137035D 03	7.080694D 04
VARIANCE:	7.433707E-04	7.329798E-01	4.556223E 02	2.475938E 05

MEAN = 1.549231E 01

VARIANCE = 9.011434E 00 STANDARD DEVIATION = 3.001905E 00

SKEWNESS = -4.122369E-03

KURTOSIS = 2.840780E 00

QUANTILES OF THE DISTRIBUTION:

ALPHA	ALPHA-QUANTILE	VARIANCE	NORMALIZED QUANTILE
0.001	6.525058E 00	5.840706E-02	-3.023095
0.002	7.202499E 00	2.115074E-02	-2.794908
0.005	7.921307E 00	1.086465E-02	-2.552787
0.010	8.647766E 00	4.937734E-03	-2.308087
0.020	9.385397E 00	3.008976E-03	-2.059627
0.025	9.645103E 00	3.421807E-03	-1.972147
0.050	1.053542E 01	1.831021E-03	-1.672254
0.100	1.158902E 01	4.436225E-03	-1.317364
0.200	1.292809E 01	1.696679E-03	-0.866317
0.300	1.389270E 01	2.344738E-03	-0.541399
0.400	1.472587E 01	2.145376E-03	-0.260755
0.500	1.549525E 01	1.542330E-03	-0.001599
0.600	1.626398E 01	9.271319E-04	0.257336
0.700	1.709274E 01	1.708522E-03	0.536495
0.800	1.806477E 01	2.143186E-03	0.863912
0.900	1.936656E 01	1.378786E-03	1.302402
0.950	2.043901E 01	2.398907E-03	1.663643
0.975	2.133118E 01	3.721221E-03	1.964158
0.980	2.160959E 01	6.842077E-03	2.057937
0.990	2.232549E 01	1.322764E-02	2.299078
0.995	2.305215E 01	2.466045E-02	2.543847
0.998	2.377327E 01	4.277430E-02	2.786746
0.999	2.423074E 01	5.857401E-02	2.940840

SIMULATION RESULTS FOR K= 0.50 AND SAMPLE SIZE = 30

THE FIRST FOUR MOMENTS ARE:

	1.549263D 01	2.448627D 02	3.943646D 03	6.465488D 04
VARIANCE:	8.401840E-04	8.840271E-01	5.544180E 02	2.888450E 05

MEAN = 1.549263E 01

VARIANCE = 4.841497E 00 STANDARD DEVIATION = 2.200340E 00

SKEWNESS = 5.866662E-03

KURTOSIS = 2.899539E 00

QUANTILES OF THE DISTRIBUTION:

ALPHA	ALPHA-QUANTILE	VARIANCE	NORMALIZED QUANTILE
0.001	8.880285E 00	4.389128E-02	-3.059215
0.002	9.289683E 00	2.895074E-02	-2.870017
0.005	9.945611E 00	8.602623E-03	-2.566889
0.010	1.043447E 01	5.714536E-03	-2.340967
0.020	1.098809E 01	1.516437E-03	-2.085118
0.025	1.118998E 01	1.717163E-03	-1.991817
0.050	1.187312E 01	2.579421E-03	-1.676113
0.100	1.266379E 01	2.036747E-03	-1.310719
0.200	1.361239E 01	1.233325E-03	-0.872335
0.300	1.431863E 01	1.244558E-03	-0.545955
0.400	1.492431E 01	9.220671E-04	-0.266046
0.500	1.548894E 01	6.967369E-04	-0.005112
0.600	1.606097E 01	1.067861E-03	0.259247
0.700	1.666298E 01	1.074037E-03	0.537456
0.800	1.736314E 01	1.784401E-03	0.861028
0.900	1.832112E 01	3.821547E-03	1.303744
0.950	1.911623E 01	5.653307E-03	1.671191
0.975	1.979460E 01	1.046887E-02	1.984694
0.980	1.999062E 01	1.242903E-02	2.075279
0.990	2.056076E 01	1.090968E-02	2.338764
0.995	2.109485E 01	1.100856E-02	2.585587
0.998	2.173543E 01	1.518452E-02	2.881621
0.999	2.213234E 01	2.993258E-02	3.065049

SIMULATION RESULTS FOR K= 0.75 AND SAMPLE SIZE = 30

THE FIRST FOUR MOMENTS ARE:

	1.550756D 01	2.437767D 02	3.882414D 03	6.261050D 04
VARIANCE:	3.363851E-04	3.336535E-01	1.948461E 02	8.956469E 04

MEAN = 1.550756E 01

VARIANCE = 3.292473E 00 STANDARD DEVIATION = 1.814517E 00

SKEWNESS = -1.425088E-02

KURTOSIS = 2.959830E 00

QUANTILES OF THE DISTRIBUTION:

ALPHA	ALPHA-QUANTILE	VARIANCE	NORMALIZED QUANTILE
0.001	9.888675E 00	1.059276E-02	-3.142752
0.002	1.025604E 01	2.259400E-02	-2.937002
0.005	1.085379E 01	1.063573E-02	-2.602219
0.010	1.131085E 01	9.343445E-03	-2.346232
0.020	1.176631E 01	4.140753E-03	-2.091141
0.025	1.194138E 01	4.512869E-03	-1.993084
0.050	1.252072E 01	1.957431E-03	-1.668615
0.100	1.316931E 01	4.416034E-04	-1.305357
0.200	1.397234E 01	5.245046E-04	-0.855603
0.300	1.455816E 01	6.824899E-04	-0.527498
0.400	1.505430E 01	8.297206E-04	-0.249624
0.500	1.551409E 01	7.378636E-04	0.007893
0.600	1.597451E 01	5.092884E-04	0.265759
0.700	1.646059E 01	9.245065E-04	0.537999
0.800	1.704263E 01	4.539427E-04	0.863987
0.900	1.783040E 01	5.107326E-04	1.305193
0.950	1.849435E 01	5.730530E-04	1.677056
0.975	1.905101E 01	2.615929E-03	1.988825
0.980	1.919690E 01	3.075583E-03	2.070533
0.990	1.969147E 01	3.800856E-03	2.347528
0.995	2.016524E 01	4.898641E-03	2.612873
0.998	2.066853E 01	3.832733E-03	2.894755
0.999	2.104907E 01	1.654354E-02	3.107885

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     SUBROUTINE GMINIT(K,BETA)                                       C
C                                                                     C
C     ADDITIONAL ENTRY POINT:                                         C
C     GAMA(Z,IX)                                                      C
C                                                                     C
C     PURPOSE:                                                        C
C                                                                     C
C     GENERATION OF GAMMA RANDOM DEVIATES WITH SHAPE                  C
C     PARAMETER LESS THAN ONE.                                        C
C                                                                     C
C     PROGRAMMER:                                                     C
C                                                                     C
C     D.W. ROBINSON, LT, USN                                          C
C                                                                     C
C     METHOD:                                                         C
C                                                                     C
C     A MODIFICATION OF MARSAGLIA'S RECTANGLE-WEDGE-TAIL              C
C     METHOD FOR NORMAL DEVIATES IS USED.  THE GAMMA PDF              C
C     IS DECOMPOSED INTO A HEAD REGION, A NUMBER                      C
C     (DEPENDENT ON K) OF RECTANGLES AND WEDGES AND A TAIL            C
C     REGION.  THE GMINIT SECTION OF THE SUBROUTINE ALSO              C
C     SETS UP A BINARY SEARCH TREE TO BE USED FOR                     C
C     EFFICIENT SELECTION OF THE PROPER REGION DURING THE             C
C     ACTUAL GENERATING PROCESS, WHICH IS HANDLED BY THE              C
C     GAMA SECTION.                                                   C
C                                                                     C
C     DESCRIPTION OF PARAMETERS:                                      C
C                                                                     C
C     K      GAMMA DISTIRBUTION SHAPE PARAMETER (MUST BE              C
C            .GE. 0.05 AND .LT. 1.0) (REAL*4)                         C
C     BETA   GAMMA DISTRIBUTION SCALE PARAMETER (REAL*4)              C
C     IX     SEED FOR UNIFORM RANDOM NUMBER GENERATOR                 C
C     Z      RETURNED GAMMA DEVIATE (REAL*4)                          C
C                                                                     C
C     NOTE THAT THE DENSITY FUNCTION OF A GAMMA RANDOM                C
C     VARIABLE IS GIVEN BY:                                           C
C                                                                     C
C                                 K  K-1  (-X/BETA)                   C
C     F(X) = (1/BETA)  X    E           / GAMMA(K)                    C
C                                                                     C
C     THE FOLLOWING SUBROUTINES ARE CALLED:                           C
C                                                                     C
C     IGAM(K,X)          COMPUTES THE INCOMPLETE GAMMA                C
C                        FUNCTION (GAMMA CDF).                        C
C     INVGAM(K,X)        COMPUTES THE INVERSE GAMA CDF                C
C     RANDOM(IX,U,N)     RETURNS A VECTOR U OF N UNIFORM              C
C                        RANDOM NUMBERS.                              C
C     OVFLOW             SET-UP ENTRY POINT FOR RANDOM                C
C                                                                     C
C     NOTE:                                                           C
C     UNDERFLOW IS POSSIBLE WHEN K IS LESS THAN .18 AND               C
C     BECOMES MORE LIKELY AS K DECREASES.  WHEN K IS .05              C
C     THE PROBABILITY OF UNDERFLOW IS ABOUT .00013 FOR                C
C     ANY GENERATED DEVIATE.                                          C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE GMINIT (K,BETA)
C
      REAL*4     K,IGAM,INVGAM
      INTEGER*4 FIRST,TABLE,BOTTOM,END
      LOGICAL*1 USED
C
      DIMENSION X(101),H(100),P(100),Q(100),R(100),B(100)
      DIMENSION PROB(202),TABLE(202),NEXT(202),LAST(202)
      DIMENSION TEST(202),LIST(202),USED(202)
      DIMENSION RAND(2)
      EQUIVALENCE (U,RAND(1)), (V,RAND(2))
```

```
C
CC      THIS FIRST SECTION INITIALIZES CONSTANTS AND TABLES
CC      TO BE USED BY GAMA WHEN IT IS CALLED.  THE FOLLOWING
CC      ARE TO BE CALCULATED:
CC
CC      FIRST   STARTING POINT FOR BINARY SEARCH
CC      NEXT,   LINK VECTORS FOR BINARY SEARCH PROCEDURE
CC       LAST
CC      PROB    VECTOR OF CUMULATIVE PROBABILITIES
CC      TABLE   USED TO LOOK-UP RESULT OF SEARCH IN PROB
CC      J1      POSITION IN PROB OF P0
CC      X(I)    LEFT-HAND BOUNDARY OF I-TH RECTANGLE
CC      H(I)    WIDTH OF I-TH RECTANGLE
CC      R(I)    REJECTION RATIO FOR I-TH WEDGE
CC      B(I)    Y ORDINATE FOR I-TH WEDGE
CC      ALPHA   K-1
CC      P0      PROBABILITY FOR HEAD REGION
CC      PN      PROBABILITY FOR TAIL REGION
CC      H1 TO   CONSTANTS FOR HEAD APPROXIMATION TO INVERSE
CC       H4     GAMMA CDF
CC
CC
C       CHECK FOR K IN RANGE
        IF((K .GE. 0.05) .AND. (K .LT. 1.0)) GO TO 5
        WRITE(6,4)K
    4 FORMAT(//'0GMINIT CALLED WITH K=',1PE16.6,
     *          ' OUT OF RANGE'/)
        RETURN
C
CC      GET UPPER BOUND ON NUMBER OF RECTANGLES REQUIRED
C
    5 N = 20. + 6.6/K
        IF(N .GT. 100) N = 100
C
CC      INITIALIZE CONSTANTS
C
        M = 2*N + 2
        MM = M - 1
        ALPHA = K - 1.
        GK = GAMMA(K)
        P0 = 5.0E-5 / (K * K)
        HFAC = 2.
C
CC      FIND CONSTANTS FOR HEAD APPROXIMATION
C
        H1 = K * GK
        H2 = 1. / K
        H3 = 2.2E-7 * (K + 1.)
        H4 = 4. / (K + 1.)
C
CC      SET UP RECTANGLE BOUNDS
C
        X(1) = (H1 * P0) ** H2
        X(1) = 2. * X(1) / (1. + SQRT(1. - H4 * X(1)))
        P0 = IGAM(K,X(1))
        H1 = H1 * P0
        H(1) = .25 * X(1)
        DO 10 I=2,N
             X(I) = X(I-1) + H(I-1)
             H(I) = H(I-1) * HFAC
             P(I) = 0.
             Q(I) = 0.
   10 CONTINUE
        X(N+1) = X(N) + H(N)
C
CC      ZERO PROBABILITY VECTORS AND LINKS
C
        DO 15 I=1,M
             NEXT(I) = 0
             LAST(I) = 0
             PROB(I) = 0
             LIST(I) = 0
```

```
            USED(I) = .FALSE.
   15 CONTINUE
C
C     FIND COEFFICIENTS FOR NEWTON-RAPHSON APPROXIMATION
C     TO DENSITY FUNCTION TANGENT
C
      B1 = -2. * ALPHA
      B2 = ALPHA * (ALPHA - 1.)
      A1 = B1 + 1.
      A2 = ALPHA * (ALPHA - 2.)
      C  = 1. - ALPHA
C
C     FIND RECTANGLE PROBABILITIES AND WEDGE VALUES
C
      PL = P0
      FL = EXP( ALPHA * ALOG(X(1)) - X(1)) / GK
      DO 40 I=1,N
           FU = EXP( ALPHA * ALOG(X(I+1)) - X(I+1)) / GK
           PU = IGAM(K,X(I+1))
           P(I) = H(I) * FU
           Q(I) = PU - PL - P(I)
C
C          NEWTON-RAPHSON ITERATION TO FIND POINT WHERE
C          TANGENT TO PDF IS PARALLEL TO CHORD
C
           W = X(I)
           S = (FU - FL) / H(I)
           SC = S * GK
           DO 20 J=1,15
                Y = W * ((W + A1) * W + A2 +
     *                SC * EXP(C * ALOG(W) + W))
     *                     / ((W + B1) * W + B2)
                IF(ABS(Y - W) .LT. 1.E-4 * Y) GO TO 30
                W = Y
   20      CONTINUE
C
C          FIND VALUES FOR REJECTION METHOD
C
   30      B(I) = EXP(ALPHA * ALOG(Y) - Y)/GK + S*(X(I) - Y)
           R(I) = (B(I) - FU) / (FL - FU)
C
C          TEST TO SEE IF ENOUGH RECTANGLES HAVE BEEN TAKEN
C
           IF(PU .GT. 0.999) GO TO 45
C
C          RESET PROBABILITY AND FUNCTION VALUES FOR
C          NEXT RECTANGLE
C
           PL = PU
           FL = FU
   40 CONTINUE
C
C     FIND LOWER END INDEX FOR PROB
C
   45 LOW = 2 * (N-I) + 1
C
C     FIND TAIL PROBABILITY
C
      PN = 1. - PU
C
C     GENERATE PROBABILITY VECTOR
C
      DO 80 I=1,N
           PROB(I) = P(I)
           PROB(I+N) = Q(I)
           TABLE(I) = I
           TABLE(I+N) = -I
   80 CONTINUE
      PROB(M-1) = P0
      PROB(M) = PN
      TABLE(M-1) = 0
      TABLE(M) = 0
```

```
C
C       SORT PROBABILITY VECTOR
C
        DO 110 I=1,MM
             ICH = 0
             L = M - I
             DO 100 J=1,L
                  IF(PROB(J) - PROB(J+1)) 100,100,90
   90             TEMP = PROB(J)
                  PROB(J) = PROB(J+1)
                  PROB(J+1) = TEMP
                  ITEMP = TABLE(J)
                  TABLE(J) = TABLE(J+1)
                  TABLE(J+1) = ITEMP
                  ICH = 1
  100        CONTINUE
             IF(ICH) 120,120,110
  110   CONTINUE
C
C       CONVERT PROB TO CUMULATIVE PROBABILITIES
C       FIND FIRST AND J1
C
  120   J1 = 0
        FIRST = 0
        L = LOW + 1
        DO 130 I=L,M
             IF((TABLE(I) .EQ. 0) .AND. (PROB(I) .EQ. P0))J1=I
             PROB(I) = PROB(I) + PROB(I-1)
             IF((PROB(I) .GE. .5) .AND. (FIRST .EQ. 0))FIRST=I
  130   CONTINUE
        IF(FIRST .EQ. M) FIRST = MM
        PROB(M) = 1.
        IF(J1 .EQ. 0) J1 = LOW
C
C       NOW DETERMINE THE VECTORS NEXT AND LAST
C
  150   BOTTOM = 1
        END = 1
        PR = .25
        LIST(1) = FIRST
        TEST(1) = .5
        USED(FIRST) = .TRUE.
C
C       FIND NEXT(K) AND LAST(K) FOR EACH K ON LIST
C
  151   DO 159 I=1,BOTTOM
             LI = LIST(I)
C
C            FIND NEXT(LI)
C
C            IF LI+1 HAS BEEN TESTED GO ON TO FIND LAST(LI)
C
             IF(USED(LI+1)) GO TO 155
C
C            GET PROBABILITY VALUE FOR NEXT(LI)
C
             PRN = TEST(I) + PR
C
C            FIND J SUCH THAT PROB(J) > PRN
C
             DO 152 J=LOW,MM
                  IF(PROB(J) .GT. PRN) GO TO 153
  152        CONTINUE
C
C            GET K SUCH THAT PROB(K) HAS NOT BEEN TESTED AND
C            LI .LT. K .LE. J.  IF K EXISTS, SET NEXT(LI)=K.
C
  153        IF( .NOT.USED(J)) GO TO 154
             J = J - 1
             IF(LI - J) 153,155,155
  154        NEXT(LI) = J
C
```

```
C           NEXT(LI) HAS NOW BEEN FOUND IF IT IS NON-ZERO.
C           NOW ADD IT TO THE LIST IN PLACE OF LI FOR THE
C           NEXT PASS THROUGH THE LIST.
C
  155       LIST(I) = NEXT(LI)
C
C           NOW FIND LAST(LI)
C
C           IF LI = LOW OR PROB(LI-1) HAS BEEN USED, GO
C           ON TO THE NEXT VALUE ON THE LIST
C
            IF((LI .EQ. LOW) .OR. USED(LI-1)) GO TO 159
C
C           GET PROBABILITY VALUE FOR LAST(LI)
C
            PRL = TEST(I) - PR
C
C           RESET PROBABILITY TEST VALUE FOR NEXT PASS
C
            TEST(I) = PRN
C
C           FIND J SUCH THAT PROB(J) > PRL
C
            DO 156 J=LOW,MM
                 IF(PROB(J) .GT. PRL) GO TO 157
  156       CONTINUE
C
C           FIND K SUCH THAT PROB(K) HAS NOT BEEN TESTED AND
C           LI .GT. K  .GE. J.  IF K EXISTS, SET LAST(LI)=K.
C
  157       IF( .NOT.USED(J)) GO TO 1571
            J = J + 1
            IF(LI - J) 158,158,157
 1571       LAST(LI) = J
            GO TO 1585
  158       J = LI
C
C           LAST(LI) HAS NOW BEEN FOUND. ADD IT TO THE END OF
C           THE LIST FOR THE NEXT PASS.
C
 1585       END = END + 1
            LIST(END) = J
            TEST(END) = PRL
  159 CONTINUE
C
C     NOW RESET THE LIST FOR THE NEXT PASS BY ELIMINATING
C     ZERO ENTRIES.
C
      BOTTOM = END
      PR = PR * .5
      I = 1
C
C     TEST FOR ZERO ENTRY
C
 1591 IF(LIST(I)) 160,160,163
C
C     ZERO ENTRY FOUND. ADJUST BOTTOM OF LIST
C
  160 BOTTOM = BOTTOM - 1
C
C     IF LIST EMPTY, QUIT.
C
      IF(BOTTOM) 165,165,161
C
C     SHIFT OTHER LIST ENTRIES UP
C
  161 DO 162 J=I,BOTTOM
            LIST(J) = LIST(J+1)
            TEST(J) = TEST(J+1)
  162 CONTINUE
      GO TO 1591
C
```

```
C       SET USED FLAG FOR LIST ENTRY
C
  163 USED(LIST(I)) = .TRUE.
C
C       GET NEXT LIST ENTRY
C
        I = I + 1
        IF (I .LE. BOTTOM) GO TO 1591
C
C       DONE WITH LIST RESETTING.  GO BACK FOR ANOTHER PASS
C
        END = BOTTOM
        GO TO 151
C
C       SETUP FIRST CALL TO RANDOM
C
  165 CALL OVFLOW
C
C       THE FOLLOWING STATEMENTS PRINT OUT A TABULAR LSTING
C       OF THE RESULTS OF GMINIT.  THEY MAY BE REMOVED, IF
C       DESIRED, IN A PRODUCTION RUN.
C
        WRITE(6,170)K,BETA
  170 FORMAT('1GENERATED VALUES FOR K=',1PE14.6,
     *       '   BETA=',E14.6/)
        WRITE(6,175)PO,PN
  175 FORMAT('0HEAD PROBABILITY=',1PE14.6,
     *       '   TAIL PROBABILITY=',E14.6//)
        WRITE(6,180)
  180 FORMAT('0RECTANGLE/WEDGE VALUES'//2X,'I',9X,'X(I)',
     *       12X,'P(I)',12X,'Q(I)',12X,'R(I)',12X,'B(I)'/)
C
C       ADD UP TOTAL RECTANGLE AND WEDGE PROBABILITIES AND
C       PRINT OUT RESULTS
C
        SUM1 = 0.
        SUM2 = 0.
        DO 192 I=1,N
C
C            NO ACTION FOR ZERO PROBABILITIES
C
             IF(P(I)) 193,193,185
C
  185        SUM1 = SUM1 + P(I)
             SUM2 = SUM2 + Q(I)
             WRITE(6,190) I,X(I),P(I),Q(I),R(I),B(I)
  190        FORMAT(1XI3,1P5E16.6)
  192 CONTINUE
C
  193 WRITE(6,194)SUM1,SUM2
  194 FORMAT('0SUMS',15X,1P2E16.6)
C
C       WRITE OUT CONSTANTS
C
        WRITE(6,195)J1,H1,H2,H3,H4
  195 FORMAT(/'0VALUES FOR HEAD/TAIL APPROXIMATION:'/
     *       ' J1=',I4/' H1=',E16.6,'   H2=',E16.6,
     *       '   H3=',E16.6,'   H4=',E16.6)
C
C       WRITE OUT BINARY SEARCH DATA
C
        WRITE(6,196)FIRST
  196 FORMAT(/'0STARTING POINT FOR BINARY SEARCH',I4)
        WRITE(6,197)(I,PROB(I),TABLE(I),NEXT(I),LAST(I),
     *                 I=LOW,M)
  197 FORMAT(/'0PROBABILITY VECTOR AND LINKS'//
     *       14X,'PROB',9X,'TABLE',4X,'NEXT',5X,
     *       'LAST'//(1X115,1PE16.6,0P3I9))
C
C       FINISHED WITH SET-UP PHASE.  QUIT.
C
        RETURN
```

```
C
C
C       THIS IS THE SECTION WHICH ACTUALLY GENERATES THE
C       GAMMA VARIATES.
C
        ENTRY  GAMA(IX,Z)
C
C       GET TWO UNIFORM RANDOM VARIATES
C
        CALL RANDOM(IX,U,2)
C
C       CONDUCT BINARY SEARCH USING THE FIRST UNIFORM VARIATE
C
        J = FIRST
  200   IF(U - PROB(J)) 210,250,230
C
C       U < CURRENT VALUE.  USE LAST FOR FOLLOWING TEST.
C
  210   IF(LAST(J)) 250,250,220
  220   J = LAST(J)
        GO TO 200
C
C       U > CURRENT VALUE.  USE NEXT FOR FOLLOWING TEST.
C
  230   IF(NEXT(J)) 245,245,240
  240   J= NEXT(J)
        GO TO 200
C
  245   J= J + 1
C
C       LOCATED PROPER PROBABILITY DIVISION.  LOOK UP IN TABLE
C
  250   N = TABLE(J)
        IF(N) 260,290,320
C
C       TABLE VALUE < 0.  SAMPLE FROM N-TH WEDGE.
C
  260   N= -N
C
C       GET ONE MORE UNIFORM DEVIATE
C
        CALL RANDOM(IX,U,1)
C
C       GET U .LE. V
C
  270   IF(U .LE. V) GO TO 280
        TEMP = U
        U = V
        V = TEMP
C
C       GET TRIAL GAMMA VALUE
C
  280   Z = X(N) + H(N)*U
C
C       PERFORM REJECTION TEST
C
        IF(V .LE. R(N)) GO TO 330
C
C       FIRST TEST FAILED.  GET VALUE FOR SECOND TEST.
C
        W = U + EXP(ALPHA * ALOG(Z) - Z) / B(N)
        IF(V .LE.W) GO TO 330
C
C       SECOND TEST ALSO FAILED.  REPEAT LOOP WITH TWO
C       NEW UNIFORM VARIATES
C
        CALL RANDOM(IX,U,2)
        GO TO 270
C
C
C       TABLE VALUE = 0.  USE HEAD/TAIL DISTRIBUTIONS.
C
```

```
C       DETERMINE WHICH DISTRIBUTION TO USE.
C
  290 IF(J .EQ. J1) GO TO 300
C
C       USE TAIL DISTRIBUTION.
C
        Z = INVGAM(K, PN * V)
        GO TO 330
C
C       USE HEAD DISTRIBUTION.
C
  300 Z = (H1 * V) ** H2
        IF(Z .LT. H3) GO TO 330
        Z = 2. * Z / (1. + SQRT(1. - H4 * Z))
        GO TO 330
C
C
C       TABLE VALUE > 0.  SAMPLE FROM N-TH RECTANGLE.
C
  320 Z = X(N) + H(N) * V
C
C
C       Z IS NOW A UNIT GAMMA VARIATE.  SCALE IT AND EXIT.
C
  330 Z = Z * BETA
        RETURN
        END
```

```
GAMA     CSECT
         ENTRY  START
BASE1    EQU    2
BASE2    EQU    3
SAVE     EQU    4
REGA     EQU    5
REGB     EQU    6
REGC     EQU    8
SRCH     EQU    7
*
*
         STM    14,REGC,12(13)    SAVE CALLING PROGRAM REGS
         BALR   BASE1,0           SET UP BASE REGISTERS
         USING  *,BASE1
         LA     BASE2,DATA
         USING  DATA,BASE2
         LR     SAVE,13           SAVE ADDRESS OF CALLING
*                                   PROGRAM SAVE AREA
         LA     13,SVAREA         LOAD OWN SAVE AREA ADDRESS
         ST     13,8(0,SAVE)      STORE OWN SAVE AREA ADDRES
         ST     SAVE,4(0,13)      STORE CALLING SAVE AREA
*                                   ADDRESS
         LA     1,ARG1            LINK TO RANDOM FOR TWO
         L      15,=V(RANDOM)       UNIFORM DEVIATES
         BALR   14,15
*
         L      REGB,FIRST        START BINARY SEARCH
         L      SRCH,U            LOAD UNIFORM FOR SEARCH
         BALR   REGC,0            LOAD BRANCH ADDRESS
*
SEARCH   LR     REGA,REGB         LOAD NEXT INDEX
         C      SRCH,PROB(REGA)   COMPARE
         BC     4,LOW
         BC     8,FOUND
*
HIGH     L      REGB,NEXT(REGA)   GET NEXT INDEX FROM "NEXT"
         BCTR   REGB,REGC         BRANCH BACK IF INDEX
*                                   IS NON-ZERO
         LA     REGA,4(REGA)
         BC     15,FOUND          THROUGH WITH SEARCH
*
LOW      L      REGB,LAST(REGA)   GET NEXT INDEX FROM "LAST"
         BCTR   REGB,REGC         BRANCH BACK IF NON-ZERO
*
FOUND    L      REGB,TABLE(REGA)  FIND WHICH METHOD TO USE
         LTR    REGB,REGB
         BC     2,RECT            RECTANGLE IF TABLE
*                                   VALUE GREATER THAN 0
*
         STD    2,SAVED           SAVE FLOATING POINT REG 2
         BC     8,TAIL            USE HEAD/TAIL IF TABLE
*                                   VALUE = 0
*
WEDGE    LCR    REGB,REGB         USE WEDGE METHOD IF TABLE
*                                   VALUE < 0
*
         LA     1,ARG2            GET NEW RANDOM NUMBER
         L      15,=V(RANDOM)
         BALR   14,15
*
W1       LE     0,U               GET U LESS THAN OR EQUAL V
         CE     0,V
         BC     12,W2
*
         LE     2,V               EXCHANGE U AND V
         STE    0,V
         LER    0,2
         STE    0,U
*
W2       ME     0,H(REGB)         FIND Z = X(N) + U * H(N)
         AE     0,X(REGB)
*
```

```
         LE      2,V             TEST IF V IS LESS THAN OR
         CE      2,R(REGB)         EQUAL TO R(N)
         BC      12,DONE         PASS REJECTION TEST IF SO
         STE     0,Z             THIS STEP IS REACHED ONLY
*                                  1/3 OF THE TIME
         LA      1,ARG3          FIND ALOG(Z)
         L       15,=V(ALOG)
         BALR    14,15
*
         ME      0,ALPHA
         SE      0,Z
         STE     0,ZP            ZP = (K - 1) * ALOG(Z) - Z
*
         LA      1,ARG3A         FIND  EXP(ZP) =
         L       15,=V(EXP)        EXP(-Z) * (Z ** K-1)
         BALR    14,15
*
         DE      0,B(REGB)       FIND W = U +
         AE      0,U               EXP(-Z)*(Z ** K-1)/B(N)
         CE      0,V             TEST IF V IS LESS THAN OR
         LE      0,Z               EQUAL TO W
         BC      2,DONE          QUIT IF SO
*
         LA      1,ARG1          OTHERWISE GET TWO NEW
         L       15,=V(RANDOM)     UNIFORM DEVIATES
         BALR    14,15
         BC      15,W1           AND REPEAT THE LOOP
*
TAIL     C       REGA,J1         SEE IF HEAD OR TAIL IS
         BC      8,HEAD            REQUIRED
         LE      0,PN            TAIL WILL BE USED.
         ME      0,V             Z = INVGAM(K, PN*V)
         STE     0,Z
         LA      1,ARG4
         L       15,=V(INVGAM)
         BALR    14,15
         BC      15,DONE
*
HEAD     LE      0,V             COMPUTE FIRST ORDER APPROX
         ME      0,H1              (V * P0 * GAMMA(K+1) )
         STE     0,Z               ** 1/K
         LA      1,ARG5
         L       15,=V(FRXPR#)
         BALR    14,15
         CE      0,H3            SEE IF SECOND ORDER APPROX
         BC      4,DONE            IS NECESSARY
         STE     0,Z
         ME      0,H4            Z = 2*Z / (1 +
         LE      2,ONEFL           SQRT(1 - 4*Z/(K+1)) )
         SER     2,0
         STE     2,H5
         LA      1,ARG6
         L       15,=V(SQRT)
         BALR    14,15
         AE      0,ONEFL
         LE      2,Z
         ME      2,TWOFL
         DER     2,0
         LER     0,2
*
DONE     LD      2,SAVED         RESTORE FLOATING POINT
*                                  REGISTER 2
         BC      15,SCALE
*
RECT     LE      0,H(REGB)       Z = X(N) + V * H(N)
         ME      0,V
         AE      0,X(REGB)
*
SCALE    ME      0,BETA          SCALE VARIATE AND EXIT
*
         LR      13,SAVE         RESTORE OLD SAVE AREA
*                                  ADDRESS
```

```
         LM      2,REGC,28(13)     RESTORE OLD GENERAL
*                                    PURPOSE REGISTERS
         L       14,12(13)         RESTORE RETURN ADDRESS
         MVI     12(13),X'FF'      SET END OF SUBROUTINE FLAG
         BCR     15,14
*
         DROP    BASE1
         DROP    BASE2
*
*          ENTRY POINT FOR INITIALIZING ROUTINE
*
START    STM     14,REGC,12(13)    SAVE GP REGISTERS
         BALR    BASE1,0           SET UP BASE REGISTERS
         USING   *,BASE1
         LA      BASE2,DATA
         USING   DATA,BASE2
         LR      SAVE,13
         LA      13,SVAREA
         ST      13,8(0,SAVE)
         ST      SAVE,4(0,13)
         L       REGA,0(1)         GET SET UP ARGUMENTS
         MVC     K(4),0(REGA)      K  (SHAPE PARAMETER)
         L       REGA,4(1)
         MVC     BETA(4),0(REGA)   BETA (SCALE PARAMETER)
         L       REGA,8(1)
         MVC     IX(4),0(REGA)     IX  (RANDOM NUMBER SEED)
*
         LA      1,ARG7            CALL GMINIT
         L       15,=V(GMINIT)
         BALR    14,15
*
         L       15,=V(OVFLOW)     CALL OVFLOW TO SET UP FOR
         BALR    14,15               RANDOM
*
         LR      13,SAVE
         LM      2,REGC,28(13)
         L       14,12(13)
         MVI     12(13),X'FF'
         BCR     15,14
*
*
         DS      0D
IX       DC      4X'00'
U        DC      4X'00'
V        DC      4X'00'
Z        DC      4X'00'
ZP       DC      4X'00'
ONE      DC      F'1'
TWO      DC      F'2'
ONEFL    DC      E'1.0'
TWOFL    DC      E'2.0'
*
K        DC      4X'00'
BETA     DC      4X'00'
*
ALPHA    DC      4X'00'
FIRST    DC      4X'00'
PROB     DC      4X'00'
P1       DC      808X'00'
NEXT     DC      4X'00'
N1       DC      808X'00'
LAST     DC      4X'00'
L1       DC      808X'00'
TABLE    DC      4X'00'
T1       DC      808X'00'
*
DATA     DS      0F
X        DC      4X'00'
X1       DC      404X'00'
H        DC      4X'00'
HP       DC      400X'00'
R        DC      4X'00'
```

```
R1          DC      400X'00'
B           DC      4X'00'
B1          DC      400X'00'
J1          DC      4X'00'
PN          DC      4X'00'
H1          DC      4X'00'
H2          DC      4X'00'
H3          DC      4X'00'
H4          DC      4X'00'
H5          DC      4X'00'
*
ARG1        DC      A(IX)
            DC      A(U)
            DC      X'80'
            DC      AL3(TWO)
ARG2        DC      A(IX)
            DC      A(U)
            DC      X'80'
            DC      AL3(ONE)
ARG3        DC      X'80'
            DC      AL3(Z)
ARG3A       DC      X'80'
            DC      AL3(ZP)
ARG4        DC      A(K)
            DC      X'80'
            DC      AL3(Z)
ARG5        DC      A(Z)
            DC      X'80'
            DC      AL3(H2)
ARG6        DC      X'80'
            DC      AL3(H5)
ARG7        DC      A(K)
            DC      A(ALPHA)
            DC      A(FIRST)
            DC      A(P1)
            DC      A(N1)
            DC      A(L1)
            DC      A(T1)
            DC      A(X1)
            DC      A(HP)
            DC      A(R1)
            DC      A(B1)
            DC      A(J1)
            DC      A(PN)
            DC      A(H1)
            DC      A(H2)
            DC      A(H3)
            DC      X'80'
            DC      AL3(H4)
*
SVAREA      DS      18F
SAVED       DS      3D
            END
```

```
      FUNCTION IGAM(K,X)
C
CC    PURPOSE:
C
C     EVALUATION OF THE INCOMPLETE GAMMA FUNCTION:
C                       K-1  -Y
C     INTEGRAL OF  Y    E   FROM 0 TO X DIVIDED BY
C     GAMMA(K)
C
C     METHOD:
C
C     FOR VALUES OF X LESS THAN 5 AN INFINITE SERIES
C     APPROXIMATION IS USED; FOR X GREATER THAN 5 A
C     CONTINUED FRACTION APPROXIMATION IS USED. IF X
C     IS LESS THAN OR EQUAL TO ZERO, ZERO IS RETURNED.
CC
C
      IMPLICIT REAL*8(D)
      REAL*4 IGAM,K,V(6)
      REAL*8 EPS/1.D-13/
C
C     TEST VALUE OF X
      IF(X .GT. 0.)GO TO 10
C
C     X .LE. 0. RETURN IGAM = 0.
      IGAM=0.
      RETURN
C
C     DECIDE WHICH APPROXIMATION TO USE
   10 IF(X .GT. 5.0)GO TO 50
C
C     USE INFINITE SERIES APPROXIMATION
      DX=DBLE(X)
      DK=DBLE(K)
C
C     INITIALIZE SUM AND TERM
      DTERM=DX**DK/DGAMMA(DK)
      DSUM=DTERM/DK
C
C     SUM THE SERIES
      DO 30 I=1,30
C
C     TEST FOR CONVERGENCE
      IF(DABS(DTERM)-EPS*DSUM)40,40,20
C
C     HASN'T CONVERGED.  ADD ANOTHER TERM
   20 DTERM=DTERM*(-DX)/DFLOAT(I)
      DSUM=DSUM+DTERM/(DK+DFLOAT(I))
   30 CONTINUE
C
C     SERIES CONVERGED. EXIT.
   40 IGAM=SNGL(DSUM)
      RETURN
C
CC
C     USE CONTINUED FRACTION APPROXIMATION
C
C     INITIALIZE CONTINUED FRACTION VALUES
   50 Y=1.-K
      W=X+Y+1.0
C
C     SET UP RATIO VECTOR
      V(1)=1.0
      V(2)=X
      V(3)=X+1.0
      V(4)=W*X
C
      R=V(3)/V(4)
      COUNT=1.0
C
C     GET NEXT APPROXIMATION
   60 Y=Y+1.0
```

```
      W=W+2.0
      V(5)=V(3)*W-V(1)*Y*COUNT
      V(6)=V(4)*W-V(2)*Y*COUNT
      RATIO=V(5)/V(6)
C
C     TEST FOR CONVERGENCE
      IF(ABS(RATIO-R) .LE. 1.0E-6*R)GO TO 90
C
C     DIDN'T CONVERGENCE.  SET UP FOR NEXT ITERATION.
      R=RATIO
      COUNT=COUNT+1.0
      DO 70 I=1,4
      V(I)=V(I+2)
   70 CONTINUE
C
C     TEST IF SCALING REQUIRED
      IF(V(5) .LT. 1.0E 35)GO TO 60
C
C     SCALE DOWN RATIO VECTOR
      DO 80 I=1,4
      V(I)=V(I)*1.0E-35
   80 CONTINUE
      GO TO 60
C
C     CONVERGED.  CONVERT RESULT AND EXIT.
   90 IGAM = 1.-(RATIO*EXP(K*ALOG(X)-X)/GAMMA(K))
      RETURN
      END
```

```
      FUNCTION  INVGAM(K,Z)
C
C     THIS FUNCTION SUBPROGRAM SOLVES THE EQUATION
C          G(K;X) = 1 - Z
C     BY NEWTON-RAPHSON ITERATION, WHERE G(K;X) IS THE
C     INCOMPLETE GAMMA FUNCTION WITH SHAPE PARAMETER K LESS
C     THAN 1.0.
C
      REAL*4 K,INVGAM,KP,KBAR
      DATA EPS/1.0E-6/
      DIMENSION V(6)
C
C     TEST FOR Z IN RANGE
      IF(Z .GT. 0.) GO TO 10
      INVGAM=0.
      RETURN
C
C     TEST IF NEW K VALUE
   10 IF(K .EQ. KP) GO TO 20
C
C     NEW VALUE.  RECOMPUTE K-DEPENDENT CONSTANTS.
      KP=K
      KBAR=1.-K
      GK=GAMMA(K)
C
C     INITIALIZE ITERATION VALUES
   20 ZBAR=-Z*GK
      X=1.0
C
C     MAIN ITERATION LOOP.
C
C     INITIALIZE CONTINUED FRACTION VALUES.
   30 Y=KBAR
      W=X+Y+1.0
      V(1)=1.0
      V(2)=X
      V(3)=X+1.0
      V(4)=W*X
      R=V(3)/V(4)
      COUNT=1.0
C
C     CONTINUED FRACTION APPROXIMATION FOR G(K;X)
   40 Y=Y+1.0
      W=W+2.0
      V(5)=V(3)*W-V(1)*Y*COUNT
      V(6)=V(4)*W-V(2)*Y*COUNT
      RATIO=V(5)/V(6)
C
C     TEST FOR CONVERGENCE
      IF(ABS(RATIO-R) .LE. EPS*R) GO TO 70
C
C     DIDN'T CONVERGE.  SET UP FOR NEXT ITERATION.
      R=RATIO
      COUNT=COUNT+1.0
      DO 50 I=1,4
      V(I)=V(I+2)
   50 CONTINUE
C
C     TEST IF SCALING REQUIRED
      IF(V(5) .LT. 1.0E 35)GO TO 40
C
C     SCALE DOWN V VECTOR TO PREVENT OVERFLOW.
      DO 60 I=1,4
      V(I)=V(I)*1.0E-35
   60 CONTINUE
      GO TO 40
C
C     APPLY NEWTON'S FORMULA
   70 XNEW=X*(1.+RATIO+ZBAR*EXP(X-KP*ALOG(X)))
C
C     TEST FOR CONVERGENCE
      IF(ABS(XNEW-X) .LE. EPS*XNEW) GO TO 80
```

```
      X=XNEW
      GO TO 30
C
C     NEWTON'S METHOD CONVERGED.  SET FUNCTION VALUE
C     AND EXIT.
   80 INVGAM=XNEW
      RETURN
      END
```

```
INVGAM   CSECT
SAVE     EQU   2                  REGISTER EQUATES
BASE     EQU   3
REGA     EQU   4
REGB     EQU   5
REGC     EQU   6
         STM   14,REGC,12(13)     SAVE GENERAL PURPOSE REGS
         LR    BASE,15            SET UP BASE REGISTER
         USING INVGAM,BASE
*
         LR    SAVE,13            SAVE CALLING PROGRAM SAVE
*                                   AREA ADDRESS
         LA    13,SVAREA          PUT OWN SAVE AREA ADDRESS
*                                   IN REGISTER 13
         ST    13,8(0,SAVE)       STORE SAVE AREA ADDRESSES
         ST    SAVE,4(0,13)
         STD   2,SAVED            SAVE FLOATING POINT REGS
         STD   4,SAVED+8
         STD   6,SAVED+16
*
         L     REGB,4(1)          GET ADDRESS OF 1 - Z
         LE    0,0(REGB)          LOAD 1 - Z
         LTER  0,0                TEST IF 1 - Z > 0.0
         BC    2,T1
         LE    0,=E'0.0'          Z .LE. 0;  SET INVGAM TO 0
*                                   AND RETURN
         LR    13,SAVE            RESTORE CALLING PROGRAM
*                                   SAVE ADDRESS
         LM    2,REGC,28(13)      RESTORE GP REGISTERS
         L     14,12(13)          RESTORE RETURN ADDRESS
         MVI   12(13),X'FF'       SET END OF SUBROUTINE FLAG
         BCR   14,15              RETURN
*
T1       STE   0,Z                SAVE VALUE OF 1 - Z
*
         L     REGB,0(1)          GET ADDRESS OF K
         LE    0,0(REGB)          LOAD VALUE OF K
         CE    0,K                TEST IF VALUE IS NEW
         BC    8,T2
         STE   0,K                VALUE IS NEW.  COMPUTE NEW
         LCER  2,0                  CONSTANTS.
         STE   2,KNEG             KNEG = -K
         AE    2,=E'1.0'
         STE   2,KBAR             KBAR = 1 - K
*
         LA    1,ARG1             GET GAMMA(K)
         L     15,=V(GAMMA)
         BALR  14,15
         STE   0,GK               GK = GAMMA(K)
*
T2       LE    0,Z                COMPUTE Z-RELATED
         LCER  0,0                  CONSTANTS
         ME    0,GK
         STE   0,ZBAR             ZBAR = -GAMMA(K) * (1 - Z)
         LE    0,=E'1.0'
         STE   0,X                X = 1
*
*            MAIN LOOP FOR NEWTON-RAPHSON ITERATION
*
LOOP1    LE    0,KBAR             INITIALIZE CONTINUED
*                                   FRACTION CONSTANTS
         STE   0,Y                Y = 1 - K
         LE    2,=E'1.0'
         STE   2,V                V(1) = 1.0
         LE    4,X
         STE   4,V+4              V(2) = X
         LER   6,4
         AER   4,2
         STE   4,V+8              V(3) = X + 1.0
         AER   0,4
         STE   0,W                W = X + Y + 1.0
         MER   0,6
```

```
         STE    0,V+12          V(4) = W * X
         DER    4,0
         STE    4,PROB          PROB = V(3) / V(4)
         STE    2,NUM           NUM = 1
*
*           CONTINUED FRACTION APPROXIMATION
*
LOOP2    LE     0,Y
         AE     0,=E'1.0'
         STE    0,Y             Y = Y + 1.0
         LE     4,W
         AE     4,=E'2.0'
         STE    4,W             W = W + 2.0
         MER    2,0
*
         LE     0,V+12
         MER    0,4
         LE     6,V+4
         MER    6,2
         SER    0,6
         STE    0,V+20          V(6) = V(4) * W -
*                                       V(2) * Y * NUM
         LE     0,V+8
         MER    0,4
         LE     6,V
         MER    6,2
         SER    0,6
         STE    0,V+16          V(5) = V(3) * W -
*                                       V(1) * Y * NUM
         LER    4,0
*
         DE     0,V+20          NEW RATIO = V(5) / V(6)
         LER    2,0
         SE     0,PROB          TEST FOR CONVERGENCE
         STE    2,PROB
         LPER   0,0
         ME     2,=E'1.0E-6'
         CER    0,2             ABS(RATIO - PROB) :
         BC     12,CNVRG                  EPS * PROB
         MVC    V(16),V+8       SHIFT RATIO VECTOR
         CE     4,=E'1.0E35'    TEST IF SCALING NEEDED
         BC     12,INCMNT
*
         LE     0,=E'1.0E-35'   SCALE DOWN RATIO VECTOR TO
         LER    2,0               PREVENT OVERFLOW
         ME     0,V
         STE    0,V
         LER    0,2
         ME     0,V+4
         STE    0,V+4
         LER    0,2
         ME     0,V+8
         STE    0,V+8
         LER    0,2
         ME     0,V+12
         STE    0,V+12
*
INCMNT   LE     2,NUM           INCREMENT COUNT
         AE     2,=E'1.0'
         STE    2,NUM
         BC     15,LOOP2
*
*
CNVRG    LA     1,ARG2          FIND ALOG(X)
         L      15,=V(ALOG)
         BALR   14,15
*
         ME     0,KNEG
         AE     0,X
         STE    0,XP            XP = -K * ALOG(X) + X
*
         LA     1,ARG3          FIND EXP(XP)
```

```
         L       15,=V(EXP)
         BALR    14,15
*
         ME      0,ZBAR          XNEW = X * (1 + RATIO -
*                                                    X -K
         AE      0,PROB                (1-Z)*GAMMA(K) E X
         AE      0,=E'1.0'       (BASIC NEWTON-RAPHSON
         ME      0,X               RECURSION RELATION)
         LER     2,0
*
         SE      2,X             TEST FOR NEWTON-RAPHSON
         LPER    2,2               CONVERGENCE
         STE     0,X
         ME      0,=E'1.0E-6'
         CER     2,0             ABS(XNEW - X) :
         BC      2,LOOP1                EPS * XNEW
*
         LE      0,X             DONE. LOAD FUNCTION VALUE.
*
         LD      2,SAVED         RESTORE FLOATING POINT
         LD      4,SAVED+8         REGISTERS
         LD      6,SAVED+16
         LR      13,SAVE         RESET OLD SAVE AREA
         LM      2,REGC,28(13)   RESTORE OLD GP REGISTERS
         L       14,12(13)       RESTORE RETURN ADDRESS
         MVI     12(13),X'FF'    SET END OF SUBROUTINE FLAG
         BCR     15,14           RETURN
DATA     DS      0D
*
SAVED    DS      3D
SVAREA   DS      18F
*
K        DC      4X'00'
Z        DC      4X'00'
*
KNEG     DC      4X'00'
KBAR     DC      4X'00'
GK       DC      4X'00'
ZBAR     DC      4X'00'
*
X        DC      4X'00'
Y        DC      4X'00'
V        DC      24X'00'
W        DC      4X'00'
PROB     DC      4X'00'
NUM      DC      4X'00'
XP       DC      4X'00'
*
ARG1     DC      X'80'
         DC      AL3(K)
*
ARG2     DC      X'80'
         DC      AL3(X)
*
ARG3     DC      X'80'
         DC      AL3(XP)
         END
```

```
      REAL*4  K,M2,MD
      REAL*8  M,R,R1,DBLE
      REAL*8  Y1,Y2
      DIMENSION  M(4),M2(4),MD(4,10)
      DIMENSION  K(4)
      DIMENSION  QUANT(23),Q2(23),QD(23,10),SCORE(23)
      DIMENSION  ALPHA(23),IQ(23)
      DIMENSION  S(10000),U(100)
      DATA       NSAMPL/10000/,SAMPL/10000./
      DATA       ALPHA/.001,.002,.005,.01,.02,.025,.05,.1,
     *                 .2,.3,.4,.5,.6,.7,.8,.9,.95,.975,.98,
     *                 .99,.995,.998,.999/
      DATA       K/.1,.25,.50,.75/
C
C     INITIALIZE CONSTANTS
      IX=13572183
      DO 10 I=1,23
      IQ(I)=SAMPL*ALPHA(I)+0.5
   10 CONTINUE
      N=30
C
C     START SIMULATION LOOP
      DO 160 INDEX=1,4
C
C     INITIALIZE GAMMA GENERATOR
      CALL START(K(INDEX),1.,IX)
C
C     FIND THEORETICAL MEAN AND VARIANCE
      TMEAN=(FLOAT(N)+1.)*.5
      TVAR=(FLOAT(N)**2-1.)/(12.*(K(INDEX)*FLOAT(N)+1.))
      TSD=SQRT(TVAR)
C
C     ZERO OUT DATA COLLECTION ARRAYS
      DO 20 I=1,4
      M(I)=0.
      M2(I)=0.
   20 CONTINUE
      DO 30 I=1,23
      QUANT(I)=0.
   30 CONTINUE
C
C     PERFORM 10 REPLICATIONS OF THE SIMULATION
      DO 90 L=1,10
C
C     GENERATE 10,000 RATIO STATISTICS
      DO 60 J=1,NSAMPL
      Y1=0.
      Y2=0.
      DO 40 I=1,N
      Y1=Y1+DBLE(GAMA(IX))
      Y2=Y2+Y1
   40 CONTINUE
C
C     FIND RATIO
      R=Y2/Y1
      S(J)=R
      R1=R
C
C     ACCUMULATE MOMENT ESTIMATORS
      DO 50 I=1,4
      M(I)=M(I)+R1
      R1=R1*R
   50 CONTINUE
   60 CONTINUE
C
C     DETERMINE MOMENT ESTIMATES FOR CURRENT SAMPLE
      DO 70 I=1,4
      MD(I,L)=M(I)/SAMPL
      M2(I)=M2(I)+MD(I,L)
      M(I)=0.
   70 CONTINUE
C
```

```
C       SORT SAMPLE AND GET QUANTILE ESTIMATES
        CALL SORT(S,1,NSAMPL)
        DO 80 I=1,23
        QD(I,L)=S(IQ(I))
        QUANT(I)=QUANT(I)+S(IQ(I))
     80 CONTINUE
     90 CONTINUE
C
C       FIND MEAN AND VARIANCE OF THE 10 MOMENT ESTIMATES
        DO 110 I=1,4
        M(I)=M2(I)/10.
        M2(I)=0.
        DO 100 J=1,10
        M2(I)=M2(I)+(M(I)-MD(I,J))**2
    100 CONTINUE
        M2(I)=M2(I)/9.
    110 CONTINUE
C
C       FIND MEAN AND VARIANCE OF THE 10 QUANTILE ESTIMATES
        DO 130 I=1,23
        QUANT(I)=QUANT(I)/10.
        Q2(I)=0.
        DO 120 J=1,10
        Q2(I)=Q2(I)+(QUANT(I)-QD(I,J))**2
    120 CONTINUE
        Q2(I)=Q2(I)/9.
        SCORE(I)=(QUANT(I)-TMEAN)/TSD
    130 CONTINUE
C
C       COMPUTE SAMPLE STATISTICS
        YMEAN=M(1)
        VAR= SAMPL*(M(2)-M(1)*M(1))/(SAMPL-1.)
        SD=SQRT(VAR)
        SKEW=(M(3)-(2.*VAR+M(2))*M(1))/(SD*VAR)
        CKURT=(((-3.*M(1)*M(1)+6.*M(2))*M(1)-4.*M(3))
     *          *M(1)+M(4))/(VAR*VAR)
C
C       PRINT OUT RESULTS
        WRITE(6,140)K(INDEX),N,M,M2,YMEAN,VAR,SD,SKEW,CKURT
    140 FORMAT('1SIMULATION RESULTS FOR K=',F5.2,' AND SAMPL',
     *          'E SIZE =',I4
     *      ///'0THE FIRST FOUR MOMENTS ARE:'//10X1P4E19.6/
     *          ' VARIANCE:',4E19.6///'0MEAN      =',E14.6/
     *          '0VARIANCE =',E14.6,4X,'STANDARD DEVIATION=',
     *          E14.6/'0SKEWNESS =',E14.6/'0KURTOSIS =',E14.6/)
C
C
        WRITE(6,150)(ALPHA(I),QUANT(I),Q2(I),SCORE(I),I=1,23)
    150 FORMAT(///'0QUANTILES OF THE DISTRIBUTION:'
     *          //5X,'ALPHA',6X,'ALPHA-QUANTILE',8X,'VARIANCE'
     *            ,3X,'NORMALIZED QUANTILE'//
     *            (3X0PF7.3,1P2E19.6,0PF14.6))
    160 CONTINUE
        STOP
        END
```

```
      SUBROUTINE  SORT(A,II,JJ)
      DIMENSION  A(JJ),IU(16),IL(16)
      INTEGER*4 A,T,TT
      M=1
      I=II
      J=JJ
    5 IF(I .GE. J) GO TO 70
   10 K=I
      IJ=(I+J)/2
      T=A(IJ)
      IF(A(I) .LE. T) GO TO 20
      A(IJ)=A(I)
      A(I)=T
      T=A(IJ)
   20 L=J
      IF(A(J) .GE. T) GO TO 40
      A(IJ)=A(J)
      A(J)=T
      T=A(IJ)
      IF(A(I) .LE. T) GO TO 40
      A(IJ)=A(I)
      A(I)=T
      T=A(IJ)
      GO TO 40
   30 A(L)=A(K)
      A(K)=TT
   40 L=L-1
      IF(A(L) .GT. T) GO TO 40
      TT=A(L)
   50 K=K+1
      IF(A(K) .LT. T)  GO TO 50
      IF(K .LE. L) GO TO 30
      IF(L-I .LE. J-K) GO TO 60
      IL(M)=I
      IU(M)=L
      I=K
      M=M+1
      GO TO 80
   60 IL(M)=K
      IU(M)=J
      J=L
      M=M+1
      GO TO 80
   70 M=M-1
      IF(M .EQ. 0) RETURN
      I=IL(M)
      J=IU(M)
   80 IF(J-I .GE. 11)GO TO 10
      IF(I .EQ. II) GO TO 5
      I=I-1
   90 I=I+1
      IF(I .EQ. J) GO TO 70
      T=A(I+1)
      IF(A(I) .LE. T) GO TO 90
      K=I
  100 A(K+1)=A(K)
      K=K-1
      IF(T .LT. A(K)) GO TO 100
      A(K+1)=T
      GO TO 90
      END
```

BIBLIOGRAPHY

1 Abramowitz, M. and Stegun, I.A., (ed.), Handbook of Mathematical Functions, pp. 260-263, Dover, 1964.

2 Ahrens, J.H. and Dieter, U., "Computer Methods for Sampling from the Exponential and Normal Distributions," Communications of the ACM, v. 15, pp. 873-882, October 1972.

3 Anderson, T.W. and Darling, D.A., "A Test of Goodness of Fit," Journal of the American Statistical Association, v. 49, pp. 765-769, December 1954.

4 Bartholomew, D.J., "Tests for Randomness in a Series of Events when the Alternative is a Trend," Journal of the Royal Statistical Society B, v. 18, pp. 234-239, 1956.

5 Cox, D.R., "Some Statistical Methods Connected with Series of Events," Journal of the Royal Statistical Society B, v. 17, pp. 129-164, 1955.

6 Cox, D.R., and Lewis, P.A.W., The Statistical Analysis of Series of Events, Methuen, 1966.

7 Cox, D.R., "Statistical Analysis of Dependencies in Point Processes," in Stochastic Point Processes, edited by P.A.W. Lewis, pp. 55-66, Wiley, 1972.

8 Cramer, H., Mathematical Methods of Statistics, Princeton University Press, 1946.

9 Fox, B.L., "Remarks about Random Sampling from Beta and Gamma Distributions," Notes on Operations Research - 5, ORC 66-13, University of California, Berkeley, April 1966.

10 Gnedenko, B.V., The Theory of Probability, Third ed, Chelsea, 1962.

11 Goodman, A.S., Lewis, P.A.W. and Robbins, H.E., "Simultaneous Estimation of Large Numbers of Quantiles in Simulation Experiments," to be published in Communications in Statistics, 1973.

12 IBM Research Report RC-1094, Statistical Distribution Programs for a Computer Language, by R.E. Bargmann and S.P. Ghosh, 22 December 1963.

13 Johnk, M.D., "Erzeugung von Betaverteilten und Gammaverteilten Zufallszahlen (Generation of Beta-distributed and Gamma-distributed Random Numbers)," Metrika, v. 8, pp. 5-15, 1964.

14 Knuth, D.E., The Art of Computer Programming, Volume II: Seminumerical Algorithms, Addison-Wesley, 1969.

15 Knuth, D.E., The Art of Computer Programming, Volume III: Sorting and Searching, Addison-Wesley, 1972.

16 Laha, R.G., "On a Characterisation of the Gamma Distribution," Annals of Mathematical Statistics, v. 25, pp. 784-787, December 1954.

17 Lewis, P.A.W., "Distribution of the Anderson-Darling Statistic," Annals of Mathematical Statistics, v. 32, pp. 1118-1124, December 1961.

18 Lewis, P.A.W., "Some Results on Tests for Poisson Processes," Biometrika, v. 52, pp. 67-78, June 1965.

19 Lewis, P.A.W., Goodman, A.S. and Miller, J.M., "A Pseudo-Random Number Generator for the System/360," IBM Systems Journal, No. 2, pp. 136-146, 1969.

20 Lewis, P.A.W. and Shedler, G.S., "Empirically Derived Micromodels for Sequences of Page Exceptions," IBM Journal of Research and Development, v. 17, pp. 86-100, March 1973.

21 Lukacs, E., "A Characterization of the Gamma Distribution," Annals of Mathematical Statistics, v. 26, pp. 319-324, June 1955.

22 Marsaglia, G., MacLaren, M.D. and Bray, T.A., "A Fast Procedure for Generating Exponential Random Variables," Communications of the ACM, v. 7, pp. 298-300, May 1964.

23 Marsaglia, G., MacLaren, M.D. and Bray, T.A., "A Fast Procedure for Generating Normal Random Variables," Communications of the ACM, v. 7, pp. 4-10, January 1964.

24 Naval Postgraduate School Technical Report NPS55LW72111A, Large Scale Computer-Aided Statistical Mathematics, by P.A.W. Lewis, November 1972.

25 Naylor, T.H., Balintfy, J.L., Burdick, D.S. and Chu, K., Computer Simulation Techniques, pp. 87-89, Wiley, 1968.

26 Phillips, D.T. and Beightler, C., "A Procedure for Generating Gamma Variates with Non-Integer Parameter Sets," Proceedings of the Fifth Annual Conference on Applications of Simulation, Boston, 1971.

27 Singleton, R.C., "Algorithm 347: An Efficient Algorithm for Sorting with Minimal Storage," Communications of the ACM, v. 12, p. 185, March

1969.

28 United States Air Force Project Rand Report R-641-PR, Generating Gamma Distributed Variates for Computer Simulation Models, by M.B. Berman, The Rand Corporation, February 1971.

29 Wald, A., and Wolfowitz, J., "Statistical Tests Based on Permutations of the Observations," Annals of Mathematical Statistics, v. 15, 1944, pp. 358-372.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Professor P.A.W. Lewis, Code 55Lw Naval Postgraduate School Monterey, California 93940	1
4.	Professor W.M. Woods, Code 55Wo Naval Postgraduate School Monterey, California 93940	1
5.	LT D.W. Robinson, USN 584A Michelson Road Monterey, California 93940	1
6.	Chairman, Computer Science Group, Code 72 Naval Postgraduate School Monterey, California 93940	1
7.	LTJG T.G. Price, USNR, Code 52Pg Naval Postgraduate School Monterey, California 93940	1

8. Professor A.F. Andrus, Code 55As 1
Naval Postgraduate School
Monterey, California 93940

9. Professor D.P. Gaver, Code 55Gv 1
Naval Postgraduate School
Monterey, California 93940

10. Mr. Gerald S. Shedler 1
IBM Research Labs, K08/282
Monterey and Cottle Roads
San Jose, California 95114

11. Chairman, Department of Operations Research 1
and Administrative Science, Code 55
Naval Postgraduate School
Monterey, California 93940

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)
Naval Postgraduate School
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

3. REPORT TITLE
The Generation of Gamma Distributed Variates and an Investigation of a Trend Test for the Gamma Renewal Process

4. DESCRIPTIVE NOTES (Type of report and, inclusive dates)
Master's Thesis; June 1973

5. AUTHOR(S) (First name, middle initial, last name)
David W. Robinson

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
June 1973	105	29

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT
Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

In testing the hypothesis that there is no monotone trend in a gamma renewal process, the use of the statistic

$$Y_J = Y_{2J} / Y_{1J},$$

where

$$Y_{1J} = \sum_{i=1}^{J} X_i$$

and

$$Y_{2J} = \sum_{i=1}^{J} S_i,$$

is investigated. The mean and variance of Y_J is developed as a function of J and it is shown that Y_J is asymptotically normal as $J \rightarrow \infty$ for the gamma renewal process. A high-speed, theoretically exact gamma pseudo-random variate generator is developed, tested and compared with other known techniques. The generator is then used to obtain the distribution of Y_J through digital computer simulation for small and moderate values of J.

DD FORM 1 NOV 68 1473 (PAGE 1)
S/N 0101-807-6811

Security Classification

A-31408

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Random Number Generators						
Gamma Distribution						
Gamma Random Variates						
Gamma Renewal Process						
Trend Tests						
Statistical Simulation Techniques						

145294
Thesis
R638
c.1
Robinson
The generation of gamma distributed variates and an investigation of a trend test for the gamma renewal process.